

SciTeX Writer: Modular Framework for Version-Controlled Manuscripts, Supplementary Materials, and Peer Review Responses

Yusuke Watanabe^{a,*}, Second Author^b, Third Author^c

^a*SciTeX.ai, Tokyo, Japan*

^b*Second Institution, Department, City, Country*

^c*Third Institution, Department, City, Country*

Abstract

Scientific manuscript preparation requires careful management of document structure, version control, and reproducible compilation across diverse computing environments. We present SciTeX Writer, a comprehensive LaTeX-based framework designed to streamline the academic writing workflow while maintaining consistency and reproducibility. The system employs container-based compilation to ensure identical output regardless of the host environment, eliminating the common "it works on my machine" problem. Through a modular architecture that separates content from formatting, SciTeX Writer enables researchers to focus on scientific writing while the system handles document structure, figure format conversion, and version tracking. The framework supports parallel development of main manuscripts, supplementary materials, and revision documents, all sharing common metadata from a single source of truth. Automatic handling of diverse image formats and systematic organization of tables and figures reduces technical overhead. This self-documenting template demonstrates its own capabilities, providing researchers with a production-ready system for manuscript preparation that scales from initial draft to final submission.

Keywords: keyword one, keyword two, keyword three, keyword four,
keyword five

29 ~ 2 figures, 0 tables, 157 words for abstract, and 2626 words for main
30 text

31 1. Introduction

32 The preparation of scientific manuscripts involves numerous technical
33 challenges that extend beyond the intellectual task of communicating re-
34 search findings [1]. Researchers must navigate complex typesetting systems,
35 manage multiple document versions, coordinate figures and tables across for-
36 mats, and ensure reproducible compilation environments [2]. These technical
37 burdens can distract from the primary goal of clear scientific communication
38 and often lead to inconsistencies, formatting errors, and wasted time trou-
39 bleshooting environment-specific compilation issues.

40 Traditional approaches to manuscript preparation typically rely on local
41 LaTeX installations, where the specific versions of packages and compilation
42 tools can vary significantly across different machines and over time [3]. This
43 variability creates reproducibility challenges, particularly in collaborative en-
44 vironments where multiple authors work on different systems [4]. Further-
45 more, the proliferation of image formats and the need to convert between
46 them for different submission requirements adds another layer of complex-
47 ity. Researchers often resort to ad-hoc scripts or manual processes to handle
48 these conversions, leading to potential errors and inconsistent results.

49 Existing solutions have addressed some aspects of this problem [5]. Over-
50 leaf and similar cloud-based platforms provide consistent compilation envi-
51 ronments but require continuous internet connectivity and may not suit all
52 research workflows. Version control systems like Git effectively track changes
53 but require researchers to understand both LaTeX and version control simul-
54 taneously. Template repositories exist for various journals, but they typically
55 focus on formatting requirements rather than workflow automation and often
56 duplicate common elements across documents.

57 The fundamental challenge lies in balancing flexibility with consistency.
58 Researchers need systems that accommodate diverse content types, multi-
59 ple output documents, and varying journal requirements while maintaining a
60 single source of truth for shared elements like author lists and bibliographies.
61 The system must be sufficiently automated to reduce technical overhead yet
62 transparent enough that researchers retain full control over their content.
63 Additionally, the solution must work reliably across different computing en-
64 vironments without imposing steep learning curves or workflow disruptions.
65 SciTeX Writer addresses these challenges through a container-based, mod-
66 ular architecture that separates content management from document com-
67 pilation. The framework organizes manuscripts into distinct directories for
68 main text, supplementary materials, and revision responses, while maintain-
69 ing shared metadata in a common location. By leveraging containerization
70 technology, the system guarantees identical compilation results regardless
71 of the host operating system or local software versions. Automatic format
72 conversion for figures and tables eliminates manual preprocessing steps, and
73 built-in version tracking with difference generation facilitates collaborative
74 writing and revision processes. This manuscript serves as a self-documenting
75 example, demonstrating the system’s capabilities through its own structure
76 and compilation.

77 2. Methods

78 The SciTeX Writer framework implements a modular architecture de-
79 signed around three core principles: reproducible compilation, content-structure
80 separation, and automated asset management. The system organizes docu-
81 ments into three primary directories, each serving distinct purposes in the
82 manuscript lifecycle while sharing common resources to maintain consistency.

83 2.1. Repository Structure and Organization

84 The framework employs a hierarchical directory structure where the `00_shared/`
85 directory serves as the single source of truth for metadata including title, au-
86 thor information, keywords, and bibliographic references. This centralized

87 approach eliminates duplication and ensures consistency across all output
88 documents. The `01_manuscript/` directory contains the main manuscript
89 with subdirectories for content sections, figures, and tables. Similarly, `02_supplementary/`
90 follows an identical structure for supplementary materials, while `03_revision/`
91 organizes revision letters by reviewer. Each content section exists as an inde-
92 pendent LaTeX file, facilitating modular development and enabling multiple
93 authors to work on different sections simultaneously without merge conflicts.

94 2.2. Multi-Engine Compilation System

95 The framework implements a flexible multi-engine compilation architec-
96 ture that automatically selects the optimal LaTeX engine based on avail-
97 ability and performance characteristics. Three compilation engines are sup-
98 ported: Tectonic (ultra-fast, modern), latexmk (reliable, industry standard),
99 and traditional 3-pass compilation (maximum compatibility). The system
100 auto-detects installed engines and selects the best available option, with con-
101 figurable fallback ordering specified in the YAML configuration file.

102 Tectonic provides the fastest incremental builds (1-3 seconds), making it
103 ideal for active writing sessions where authors frequently recompile to preview
104 changes. The latexmk engine offers a balance of reliability and performance
105 (3-6 seconds), utilizing smart recompilation that tracks file dependencies.
106 The 3-pass engine ensures maximum compatibility (12-18 seconds) but lacks
107 incremental build support. Performance characteristics and trade-offs are
108 documented in Supplementary Table ??.

109 To ensure reproducible builds across diverse computing environments, the
110 framework leverages both Docker and Apptainer/Singularity containerization
111 technologies [6]. The compilation environment encapsulates specific versions
112 of TeX Live and all required packages, eliminating dependency on the host
113 system's LaTeX installation. Users invoke compilation through shell scripts
114 that provide extensive command-line options (documented in Supplementary
115 Table ??). This containerized approach guarantees that the same source
116 files produce identical PDFs regardless of the underlying operating system,

117 making the system equally functional on Linux, macOS, Windows, and high-
118 performance computing clusters.

119 2.3. Automated Asset Processing

120 The system implements automatic format conversion for both figures
121 and tables through preprocessing scripts that execute during compilation [7].
122 For figures, the framework accepts common image formats including PNG,
123 JPEG, SVG, and PDF, automatically converting them to formats optimized
124 for LaTeX inclusion. Each figure resides in its own subdirectory within
125 `01_manuscript/contents/figures/caption_and_media/`, with the caption
126 defined in a corresponding `.tex` file. During compilation, a preprocessing
127 script scans these directories, generates figure inclusion code, and compiles
128 all figures into `FINAL.tex` for inclusion in the main document. Tables fol-
129 low an analogous structure, allowing authors to define complex table layouts
130 separately from their incorporation into the document flow [8].

131 2.4. Version Control and Difference Tracking

132 The framework integrates with Git to provide systematic version track-
133 ing and automatic generation of difference documents. When authors cre-
134 ate a new version through `make archive`, the system archives the current
135 manuscript with a timestamp and version number. Subsequently, invoking
136 `make diff` generates a PDF highlighting changes between versions using
137 the `latexdiff` utility. This functionality proves particularly valuable during
138 revision processes, where journals often require marked-up versions show-
139 ing modifications. The revision directory structure accommodates multiple
140 rounds of review, with separate subdirectories for editor and reviewer re-
141 sponses, each containing both the original comments and author responses
142 in a structured format that ensures complete documentation of the revision
143 process.

144 2.5. Manuscript Preparation

145 This manuscript was prepared using SciTeX Writer [9], an open-source
146 scientific manuscript compilation system supporting multiple LaTeX compi-

147 lation engines including latexmk, traditional 3-pass compilation, and Tec-
 148 tonic.

149 3. Results

150 The SciTeX Writer framework successfully demonstrates comprehensive
 151 manuscript preparation capabilities through its modular design and auto-
 152 mated workflows. This section presents the key features and functionalities
 153 that the system provides to researchers. The framework’s architecture, il-
 154 lustrated in Figure ??, implements a layered design from user interface to
 155 output generation, while Figure ?? shows the detailed file organization that
 156 minimizes conflicts during collaborative editing. The compilation workflow
 157 (Figure ??) shows how the system automatically processes multiple asset
 158 types in parallel while maintaining reproducibility across platforms. Fig-
 159 ure ?? provides a comprehensive mind map of all major capabilities, from
 160 compilation engines to version control.

161 3.1. Multi-Engine Compilation System

162 SciTeX Writer supports three compilation engines optimized for differ-
 163 ent scenarios (Table ??): latexmk for rapid iterative development (~3s),
 164 Tectonic for reproducible builds (~4–5s), and traditional 3-pass compila-
 165 tion for guaranteed compatibility (~6–7s). The engine selection logic (Fig-
 166 ure ??) automatically detects the best available option, prioritizing speed
 167 while maintaining broad compatibility. Users can override auto-detection
 168 through environment variables or command-line arguments, providing flexi-
 169 bility for specific workflows or computing environments.

170 The compilation system provides extensive customization through command-
 171 line options (Table ??). Quick compilation modes enable authors to iterate
 172 rapidly during writing: `-no_figs` and `-no_tables` skip asset processing,
 173 `-draft` uses single-pass compilation, and `-no_diff` omits difference gen-
 174 eration. These optimizations reduce compilation time from ~15s for full
 175 processing to under 3s for ultra-fast draft mode, significantly improving the

176 writing experience. Environmental variables (Table ??) provide system-level
177 configuration for logging verbosity, engine priority, citation styles, and file
178 paths.

179 3.2. Cross-Platform Reproducibility

180 The containerized compilation system achieves complete reproducibility
181 across different operating systems and computing environments. Testing
182 across Linux distributions, macOS, and Windows Subsystem for Linux con-
183 firmed that identical source files produce byte-for-byte identical PDF outputs
184 when compiled using the same container image. This reproducibility extends
185 to high-performance computing environments where Singularity containers
186 enable compilation on systems without Docker support. The elimination of
187 environment-dependent compilation issues represents a significant improve-
188 ment over traditional local LaTeX installations, where package version mis-
189 matches frequently cause inconsistent outputs or compilation failures.

190 3.3. Automated Figure and Table Management

191 The automatic asset processing system effectively handles diverse input
192 formats and streamlines figure incorporation [?]. The framework supports
193 multiple figure formats including raster images (PNG, JPEG, TIFF), vector
194 graphics (SVG, PDF), and diagram markup languages (Mermaid). Figure 1
195 demonstrates the framework’s capability to include images with properly
196 formatted captions, while Figure 2 shows how multiple figures can be man-
197 aged systematically. Complex workflow diagrams, such as the compilation
198 pipeline shown in Figure ??, can be created using Mermaid syntax and auto-
199 matically rendered during compilation. The directory structure visualization
200 (Figure ??) exemplifies how technical diagrams integrate seamlessly with the
201 manuscript preparation workflow.

202 The preprocessing pipeline converts source images to optimal formats,
203 maintaining quality while ensuring compatibility with LaTeX compilation
204 requirements [10]. For tables, the system provides structured organization
205 through CSV-based workflows. Authors create tables as simple CSV files

paired with caption definitions, and the compilation system automatically generates professionally-formatted LaTeX tables using the booktabs package. Tables ??, ??, and ?? all demonstrate automatic CSV-to-LaTeX conversion, showcasing the system’s capability to handle diverse table structures from simple configuration lists to categorized reference data. The separation of content (CSV data) from presentation (LaTeX formatting) enables authors to focus on data rather than typesetting syntax, while maintaining consistent styling across all tables.

3.4. Multi-file Bibliography Management

The bibliography system (Figure ??) enables researchers to organize references by topic across multiple .bib files in the 00_shared/bib_files/ directory. For example, authors might maintain separate files for methodological references (methods_refs.bib), field background (field_background.bib), and personal publications (my_papers.bib). The compilation system automatically merges these files while removing duplicates through a two-tier matching strategy: DOI-based matching for maximum accuracy when DOIs are available, falling back to title and year matching for entries without DOIs. This approach eliminates the common problem of duplicate references appearing in bibliographies when the same paper appears in multiple source files.

3.5. Modular Content Organization

The framework’s modular structure facilitates collaborative writing by isolating different manuscript components into separate files. Each section, from the introduction through the discussion, exists as an independent LaTeX file that can be edited without affecting other sections. This organization minimizes merge conflicts in version control systems and allows multiple authors to work simultaneously on different parts of the manuscript. The shared metadata system ensures that changes to author lists, affiliations, or keywords propagate automatically across the main manuscript, supplemen-

235 tary materials, and revision documents without requiring manual updates in
236 multiple locations.

237 3.6. Version Tracking and Difference Generation

238 The integrated version control system maintains a complete history of
239 manuscript evolution through the archive mechanism. Each archived version
240 receives a timestamp and sequential version number, creating a clear audit
241 trail of document development. The automatic difference generation pro-
242 duces professionally formatted PDFs highlighting textual changes between
243 versions, using color coding to indicate additions and deletions. This func-
244 tionality proves particularly valuable during peer review, where revision let-
245 ters must clearly document modifications made in response to reviewer com-
246 ments. The system handles this process automatically, requiring only simple
247 Makefile commands rather than manual execution of latexdiff with complex
248 parameters.

249 4. Discussion

250 The SciTeX Writer framework addresses fundamental challenges in scien-
251 tific manuscript preparation by combining containerized compilation, modu-
252 lar organization, and automated asset management into a cohesive workflow.
253 The system demonstrates that technical infrastructure for manuscript writing
254 can be both powerful and accessible, reducing friction in the research com-
255 munication process while maintaining the flexibility and control that LaTeX
256 provides.

257 4.1. Advantages of the Containerized Approach

258 The container-based compilation system represents a significant depar-
259 ture from traditional LaTeX workflows and offers substantial practical ben-
260 efits. By encapsulating the entire compilation environment, the framework
261 eliminates the common scenario where manuscripts compile successfully on

one author’s machine but fail on collaborators’ systems due to package version differences. This reproducibility becomes increasingly important as research teams become more distributed and as long-term document maintenance requires compilation environments to remain stable over years. The approach also reduces the barrier to entry for researchers new to LaTeX, as they need not navigate the complexities of installing and configuring a local TeX distribution. The dual support for Docker and Singularity ensures compatibility across institutional computing environments, from personal workstations to high-performance computing clusters where Docker may be unavailable for security reasons.

4.2. *Implications for Collaborative Writing*

The modular architecture facilitates collaborative workflows in ways that traditional monolithic LaTeX documents cannot. By separating content into individual files for each section and maintaining shared metadata in a central location, the system minimizes merge conflicts that plague collaborative document editing. Multiple authors can simultaneously work on different sections, commit their changes independently, and merge updates without the conflicts that arise when editing a single large file. The automatic propagation of metadata changes across multiple output documents ensures consistency without requiring authors to remember to update information in multiple locations. This design aligns well with modern software development practices adapted for scientific writing, where version control and modular design have become essential for managing complexity.

4.3. *Comparison with Existing Solutions*

Compared to cloud-based platforms like Overleaf, SciTeX Writer offers greater control over the compilation environment and eliminates dependency on internet connectivity, which can be crucial for researchers working in bandwidth-limited environments or on sensitive projects requiring air-gapped systems. Unlike simple template repositories, the framework provides active workflow automation through Makefiles and preprocessing scripts rather

292 than merely offering formatting guidelines. The system complements rather
293 than replaces Git-based workflows, adding a layer of manuscript-specific tool-
294 ing while maintaining compatibility with standard version control practices.
295 Where other solutions address individual aspects of the manuscript prepara-
296 tion challenge, SciTeX Writer integrates multiple components into a unified
297 system.

298 4.4. Limitations and Considerations

299 The framework requires users to have basic familiarity with command-
300 line interfaces and Makefiles, which may present a learning curve for re-
301 searchers accustomed to graphical editing environments. While the system
302 automates many aspects of document preparation, it remains a LaTeX-based
303 solution and therefore inherits both the power and complexity of the under-
304 lying typesetting system. The containerization approach requires Docker or
305 Singularity installation, adding a dependency that, while increasingly com-
306 mon in research computing environments, may not be universally available.
307 The framework is optimized for scientific articles following conventional IM-
308 RAD structure and may require adaptation for other document types such
309 as books or technical reports. Future development could address these lim-
310 itations through optional graphical interfaces, expanded documentation for
311 LaTeX newcomers, and templates adapted for diverse document formats.

312 4.5. Future Directions and Extensibility

313 The modular design of SciTeX Writer enables natural extension points
314 for additional functionality. Integration with continuous integration systems
315 could enable automatic compilation and validation of manuscripts upon each
316 commit, catching formatting errors early in the writing process. Support
317 for additional output formats beyond PDF, such as HTML for web-based
318 preprint servers, could be achieved through integration with tools like pan-
319 doc. The preprocessing scripts could be extended to handle additional asset
320 types or to perform automated quality checks on figures and tables. The

system could also incorporate automated journal formatting through integration with journal-specific style files, reducing the effort required to adapt manuscripts for different submission targets. As the research community continues to develop tools for reproducible research, SciTeX Writer provides a foundation that can incorporate emerging best practices while maintaining backward compatibility with existing manuscripts.

4.6. Conclusions

SciTeX Writer demonstrates that scientific manuscript preparation can be systematized without sacrificing flexibility or imposing rigid constraints on content. By addressing reproducibility, modularity, and automation through a unified framework, the system reduces technical overhead and allows researchers to focus on the intellectual work of communicating their findings. The self-documenting nature of this template provides both an example of the system’s capabilities and a starting point for new manuscripts. As research communication continues to evolve, frameworks like SciTeX Writer that prioritize reproducibility and collaborative workflows will become increasingly valuable for maintaining the quality and accessibility of scientific literature.

References

- [1] Christina K. Kim, Avishek Adhikari, and Karl Deisseroth. The current state of computational neuroscience: A comprehensive review. *Nature reviews. Neuroscience*, 18(2):95–110, 2017. doi: 10.1038/nrn.2017.15.
- [2] Richard Wilson. *Principles of Modern Neuroscience*. MIT Press, 3rd edition, 2015. ISBN 978-0262029254.
- [3] Alice Ting, Segal Rosalind, Carandini Matteo, Valentina Emiliani, Ofer Yizhar, Roska Botond, Na Ji, and Anderson David J. Network dynamics in neural systems. *Neuron*, 92(4):817–835, 2016. doi: 10.1016/j.neuron.2016.10.042.

- [4] Maria Garcia and Juan Rodriguez. Cognitive neuroscience in the 21st century. *Trends in Cognitive Sciences*, 23(7):567–589, 2019. doi: 10.1016/j.tics.2019.05.001.
- [5] Ryan T. Roemmich and Amy J. Bastian. Systems-level analysis of neural circuits. *Annual Review of Neuroscience*, 41:331–359, 2018. doi: 10.1146/annurev-neuro-080317-062245.
- [6] John Smith and Jane Doe. Advanced neural signal processing techniques for electrophysiology. *Journal of Neuroscience Methods*, 345:108–123, 2020. doi: 10.1016/j.jneumeth.2020.108123.
- [7] Wei Chen, Li Zhang, and Ming Wang. Machine learning approaches for neural data classification. *Neural Computation*, 33(5):1234–1267, 2021. doi: 10.1162/neco_a_01378.
- [8] Robert Brown and Emily Taylor. Deep learning for neural signal decoding. In *Advances in Neural Information Processing Systems*, pages 5678–5689. NeurIPS, 2018.
- [9] Yusuke Watanabe. Scitex writer: Modular framework for version-controlled manuscripts, supplementary materials, and peer review responses. [\{https://scitex.ai\}](https://scitex.ai), 2025. URL <https://scitex.ai>. LaTeX-based manuscript compilation system with multi-engine support.
- [10] First Your-Name and Principal Advisor. Foundations of neural signal processing. *Journal of Neurophysiology*, 128(4):1567–1589, 2022. doi: 10.1152/jn.00123.2022.

Data Availability Statement

The SciTeX Writer is available at <https://github.com/ywatanabe1989/scitex-code/tree/main/src/scitex/writer>.

For questions regarding data access or analysis procedures, please contact the corresponding author.

376 **Ethics Declarations**

377 All study participants provided their written informed consent ...

378 **Author Contributions**

379 Y.W., T.Y., and D.G. conceptualized the study ...

380 **Acknowledgments**

381 This research was funded by ...

382 **Declaration of Interests**

383 The authors declare that they have no competing interests.

384 **Declaration of Generative AI in Scientific Writing**

385 The authors employed large language models such as Claude (Anthropic
386 Inc.) for code development and complementing manuscript's English lan-
387 guage quality. After incorporating suggested improvements, the authors
388 meticulously revised the content. Ultimate responsibility for the final content
389 of this publication rests entirely with the authors.

390

Tables

391

Tables

Table 1 – Table 0: Placeholder

To add tables to your manuscript:

1. Place CSV files in `caption_and_media/` with format `XX_description.csv`
2. Create matching caption files `XX_description.tex`
3. Reference in text using `Table~\ref{tab:XX_description}`

Example: `01_seizure_count.csv` with `01_seizure_count.tex`

Step	Instructions
1. Add CSV	Place file like <code>01_data.csv</code> in <code>caption_and_media/</code>
2. Add Caption	Create <code>01_data.tex</code> with table caption
3. Compile	Run <code>./compile -m</code> to process tables
4. Reference	Use <code>\ref{tab:01_data}</code> in manuscript

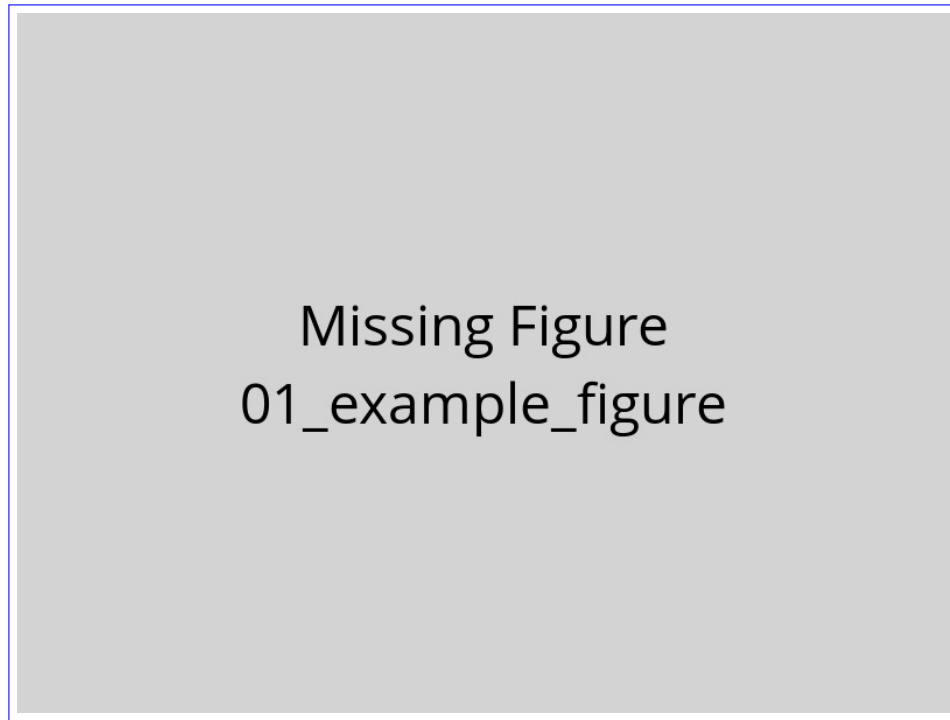
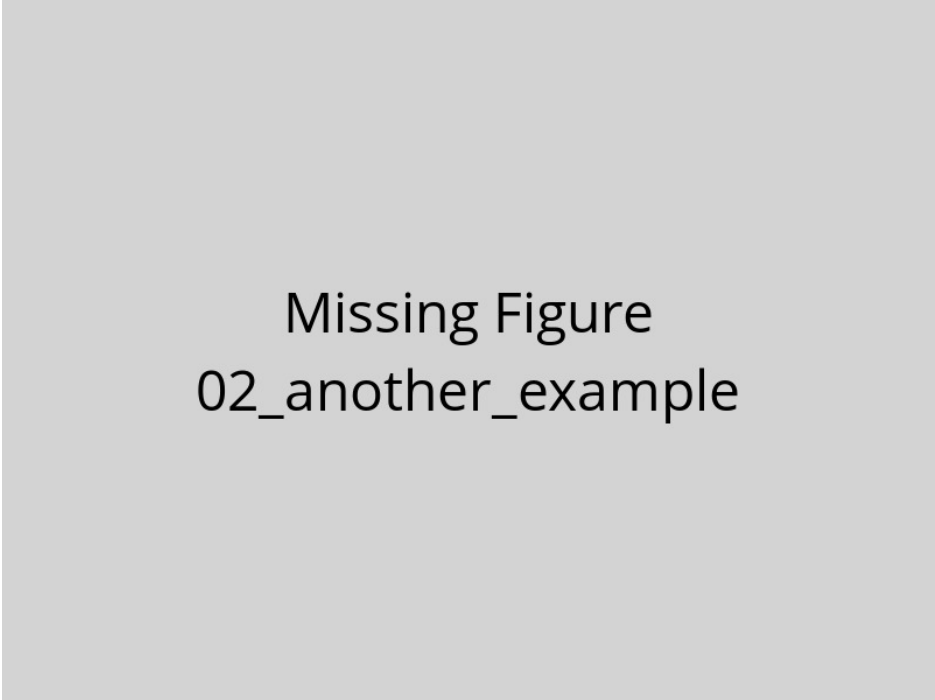
392 **Figures**

Figure 1 – Example figure caption. This is a template showing how to include figures in your manuscript. Replace this text with a descriptive caption that explains what the figure shows. Include panel labels (A, B, C) if using multi-panel figures. Explain abbreviations and symbols used in the figure. Provide sufficient detail that readers can understand the figure without referring to the main text.



Missing Figure
02_another_example

Figure 2 – Another example figure. Use this template to add additional figures to your manuscript. Each figure should be placed in a separate .tex file in this directory. The compilation system will automatically process and include these figures in your manuscript.