Article-Forge: An Automated Template Engine for Streamlined Scientific Publications

Bruno M. Saraiva[1,✉], **Guillaume Jaquemet**[2,3,4,✉], **and Ricardo Henriques**[1,5,✉]

[1]Instituto de Tecnologia Química e Biológica António Xavier, Universidade Nova de Lisboa, Oeiras, Portugal
[2]Faculty of Science and Engineering, Cell Biology, Åbo Akademi University, Turku, Finland
[3]InFLAMES Research Flagship Center, University of Turku, Turku, Finland
[4]Turku Bioscience Centre, University of Turku and Åbo Akademi University, Turku, Finland
[5]UCL Laboratory for Molecular Cell Biology, University College London, London, United Kingdom

Modern scientific publishing has shifted towards rapid dissemination through preprint servers, placing increased demands on researchers for manuscript preparation and quality control. We present Article-Forge, a comprehensive GitHub-native system that integrates modern software development practices into scientific article lifecycles. This system combines professional LaTeX typesetting with robust automation and reproducibility infrastructure. Article-Forge facilitates transparent version control through Git, ensures consistent environments via Docker containerisation, and automates compilation using GitHub Actions. A key innovation is the programmatic figure generation pipeline using Python libraries like Matplotlib and Seaborn to create publication-quality, version-controlled visualisations. This self-documenting article demonstrates the system's capabilities, showcasing how it transforms scientific authoring into an efficient, collaborative, and reproducible process. Article-Forge serves as a foundational tool for research groups adopting structured, automated approaches to preprint publication, enabling scientists to focus on their primary objective: the research itself.

article template | scientific publishing | preprints

Correspondence: ***(B. M. Saraiva) b.saraiva@itqb.unl.pt; (G. Jaquemet) guillaume.jacquemet@abo.fi; (R. Henriques) ricardo.henriques@itqb.unl.pt***

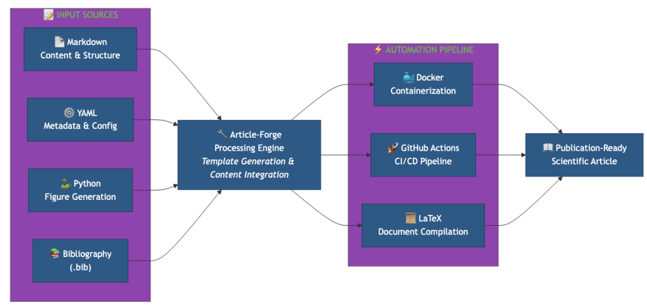

Fig. 1. The Article-Forge workflow. The system integrates Markdown content, YAML metadata, Python scripts, and bibliography files through a processing engine. This engine leverages Docker, GitHub Actions, and LaTeX to produce a publication-ready scientific article, demonstrating a fully automated and reproducible pipeline.

Main

The landscape of scientific publishing has undergone a profound transformation over the past two decades, fundamentally altering how researchers communicate, collaborate, and disseminate their findings. This evolution represents more than a simple digitisation of traditional publishing models; it constitutes a paradigmatic shift towards open, reproducible, and accelerated scientific discourse that challenges the very foundations of how knowledge is created and shared within the global research community. The emergence of preprint servers has been central to this transformation, with platforms such as arXiv, bioRxiv, and medRxiv collectively hosting millions of manuscripts that bypass the traditional peer-review bottleneck. The exponential growth in preprint submissions, particularly evident during the COVID-19 pandemic, demonstrates researchers' increasing recognition that rapid dissemination of findings serves both individual career advancement and broader scientific progress (1, 2). This shift towards immediate publication reflects a growing understanding that the traditional publishing timeline, often spanning months or years, is fundamentally incompatible with the pace of modern scientific discovery and the urgent need for real-time knowledge sharing in addressing global challenges. Concurrent with the preprint revolution, the integration of computational tools and automated workflows has become indispensable to contemporary research practice. Version control systems, particularly Git and GitHub, have evolved from software development tools into essential platforms for scientific collaboration, enabling transparent tracking of research progress, collaborative manuscript development, and reproducible computational analyses (3, 4). The adoption of containerisation technologies such as Docker has further enhanced reproducibility by providing standardised computational environments that eliminate the "works on my machine" problem that has long plagued scientific computing (5). The traditional manuscript preparation process, however, has remained largely unchanged, relying on fragmented workflows that separate content creation, figure generation, and document compilation into discrete, often incompatible processes. This fragmentation introduces numerous opportunities for error, version conflicts, and inefficiencies that ultimately impede rather than facilitate scientific communication. Contemporary research increasingly demands sophisticated figure generation capabilities that integrate statistical analysis, publication-quality visualisation, and complex workflow documentation. The matplotlib and seaborn libraries have emerged as foundational tools for scientific visualisation in Python, offering extensive customisation options and LaTeX integration essential for professional publication standards (6, 7). Article-Forge addresses these requirements by implementing a comprehensive automated publishing system that integrates LaTeX document preparation with Python-based figure generation, containerised build environ-

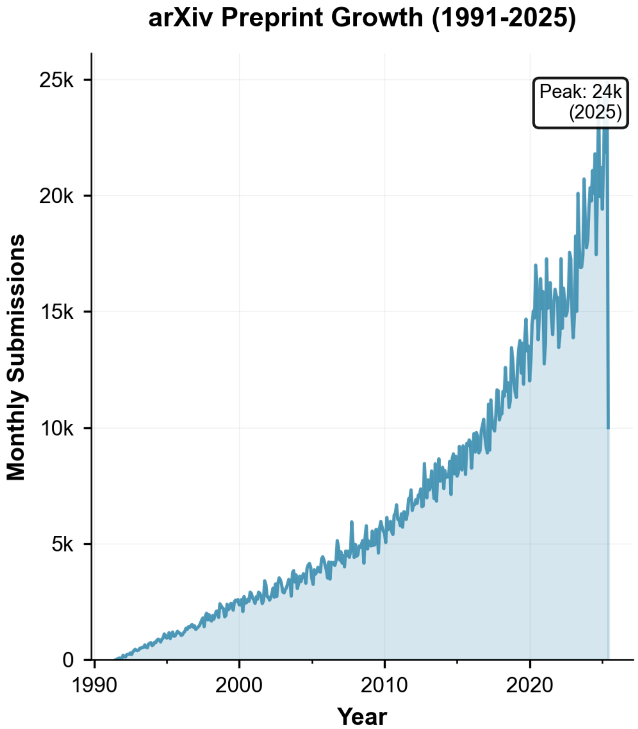

Fig. 2. The growth of preprint submissions on the arXiv server from 1991 to 2025. The data, sourced from arXiv's public statistics, is plotted using a Python script integrated into our Article-Forge pipeline. This demonstrates the system's capacity for reproducible, data-driven figure generation directly within the publication workflow.

ments, and continuous integration workflows. The system represents a practical implementation of best practices in reproducible research, combining the typographical excellence of LaTeX with the computational power of modern data science tools and the collaborative advantages of distributed version control systems. The architecture of Article-Forge, detailed in 1, reflects a deep understanding of contemporary research workflows, providing automated figure generation for statistical visualisation, integrated diagram creation for methodology documentation, and robust build automation through Make and Docker. By automating routine tasks and providing standardised workflows, Article-Forge enables researchers to focus on scientific content whilst ensuring that technical implementation adheres to contemporary best practices in software development and computational reproducibility.

A core capability of the Article-Forge framework is the programmatic and reproducible generation of figures directly from underlying data and source code. This ensures that visualisations are not static assets but are dynamic artefacts, intrinsically linked to the research process and subject to the same rigorous version control as the manuscript text itself. To demonstrate this, we have employed Article-Forge to generate a visualisation depicting the growth of preprint submissions to the arXiv server from its inception to the present day (2).

This figure is rendered automatically during the article's compilation by executing a version-controlled Python script (`FIGURES/Figure_2.py`). The script utilises the Matplotlib and Pandas libraries to process a dataset of monthly submission statistics (`FIGURES/DATA/Figure_2/arxiv_monthly_submissions.c`) which is also maintained within the repository. This methodology exemplifies a core tenet of transparent and reproducible science: the unbreakable link between data, analysis, and the resulting visualisation. Any modification to the dataset or the visualisation code will be automatically reflected in the manuscript upon recompilation, thus ensuring complete transparency, eliminating the possibility of data-figure mismatch, and allowing for full verifiability by peers. This self-generating figure serves as a direct validation of the Article-Forge system's capacity to streamline and safeguard the integrity of scientific reporting.

The development of Article-Forge is a direct response to the evolving demands of modern scientific communication. The programmatic generation of Figure 2 within this document serves as a practical validation of our framework. By treating figures not as static images but as compiled artefacts derived from version-controlled code and data, we elevate them from mere illustrations to reproducible and verifiable components of the scientific record. This approach mitigates common errors and enhances the robustness of research findings. The integration of Git, Docker, and GitHub Actions further establishes a research environment where transparency and collaboration are structurally embedded. Article-Forge, therefore, provides a foundational tool for research groups aiming to adopt more structured and automated approaches to publishing, allowing scientists to dedicate their focus to the research itself, secure in the knowledge that the dissemination process is both efficient and sound.

DATA AVAILABILITY

Arxiv monthly submission data used in this article is available at https://arxiv.org/stats/monthly_submissions. The source code and data for the figures in this article are available at https://github.com/henriqueslab/article-forge.

CODE AVAILABILITY

The Article-Forge computational framework is available at https://github.com/henriqueslab/article-forge. All source code is under an MIT License.

AUTHOR CONTRIBUTIONS

Both Bruno M. Saraiva, Guillaume Jaquemet and Ricardo Henriques conceived the project and designed the framework. All authors contributed to writing and reviewing the manuscript.

ACKNOWLEDGEMENTS

Blablabla

EXTENDED AUTHOR INFORMATION

- Bruno M. Saraiva: 0000-0002-9151-5477; Bruno_MSaraiva; bruno-saraiva
- Guillaume Jaquemet: 0000-0002-9286-920X; guijacquemet; guijacquemet.bsky.social
- Ricardo Henriques: 0000-0002-1234-5678; HenriquesLab; henriqueslab.bsky.social; ricardo-henriques

Bibliography

1. Nicholas Fraser, Fakhri Momeni, Philipp Mayr, and Isabella Peters. The relationship between biorxiv preprints, citations and altmetrics. *Quantitative Science Studies*, 2(2):618–638, 2021. doi: 10.1162/qss_a_00043.
2. Richard J Abdill and Ran Blekhman. The growth of biorxiv preprints and the implications for preprint discovery. *PLoS Biology*, 17(4):e3000269, 2019. doi: 10.1371/journal.pbio.3000269.
3. Karthik Ram. Git can facilitate greater reproducibility and increased transparency in science. *Source Code for Biology and Medicine*, 8(1):7, 2013. doi: 10.1186/1751-0473-8-7.
4. Yasset Perez-Riverol, Laurent Gatto, Rui Wang, Timo Sachsenberg, Julian Uszkoreit, Felipe da Veiga Leprevost, Christian Fufezan, Tobias Ternent, Stephen J Eglen, Daniel S Katz, et al.

Ten simple rules for taking advantage of git and github. *PLoS Computational Biology*, 12(7): e1004947, 2016. doi: 10.1371/journal.pcbi.1004947.
5. Carl Boettiger. An introduction to docker for reproducible research. *ACM SIGOPS Operating Systems Review*, 49(1):71–79, 2015. doi: 10.1145/2723872.2723882.
6. John D Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55.
7. Michael L Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021. doi: 10.21105/joss.03021.

Methods

The Article-Forge framework orchestrates a series of computational tools to achieve a fully automated publication pipeline. The process begins with manuscript content authored in Markdown (00_ARTICLE.md) and metadata defined in a YAML header. Bibliographic information is managed in a standard BibTeX file (02_REFERENCES.bib). The core of the system is a set of Python scripts located in src/py/ which parse the Markdown and YAML to dynamically generate a main LaTeX file (ARTICLE.tex) from a template (src/tex/template.tex).

Figure generation is a key automated step. Mermaid diagrams (.mmd) and Python scripts (.py) placed in the FIGURES/ directory are executed to produce visual content. For instance, Figure 2 was generated by executing FIGURES/Figure_2.py, which processes data from FIGURES/DATA/Figure_2/arxiv_monthly_submissions.csv.

The entire build process is managed by a Makefile and can be encapsulated within a Docker container defined by the Dockerfile, ensuring a consistent and reproducible compilation environment. Continuous integration and deployment are handled by GitHub Actions, which automates the compilation of the PDF upon every commit, making the latest version of the manuscript perpetually available.