Can We Standardize LaTeX? Discovering Patterns in Real-World Repositories

BART GRIEPSMA, University of Twente, The Netherlands

Despite its widespread use in academia, LaTeX lacks standardized coding conventions for organizing and formatting source files. This absence of structure can hinder collaboration, readability, and long-term maintenance of LaTeX documents. This study investigates whether consistent structural and stylistic patterns exist across LaTeX projects and explores the feasibility of developing a community-informed coding convention. By collecting and analyzing 215 publicly available GitHub repositories containing academic theses and dissertations, the research employs feature extraction and clustering methods to identify common practices. The analysis reveals several distinct usage patterns in terms of modularity, macro definition, and code formatting. These findings highlight both the diversity and recurring tendencies in LaTeX usage, suggesting that a flexible but structured style guide could be beneficial to the academic community. The study provides empirical insights that lay the groundwork for future standardization efforts.

Additional Key Words and Phrases: LaTeX, coding conventions, feature extraction, clustering, academic writing

1 INTRODUCTION

This paper investigates how LaTeX is used in practice across a wide range of publicly available repositories, with a particular focus on structural and stylistic conventions. While LaTeX has long been the de facto standard for typesetting academic and technical documents, there is little systematic research on how users actually structure their LaTeX source code in real-world projects.

The goal of this study is to identify common practices in LaTeX projects. Through automated collection and feature extraction from LaTeX repositories hosted on GitHub, this work seeks to shed light on the coding practices behind LaTeX documents—practices that are often invisible in the final typeset output but significantly impact collaboration, maintainability, and reproducibility.

By clustering repositories based on structural features, this research aims to provide a clearer understanding of how LaTeX is used across different user groups and project types. The findings may inform future efforts to develop style guidelines, editor support tools, or conventions that promote consistency and best practices in LaTeX authoring.

2 BACKGROUND

LaTeX is a document preparation system widely adopted in academic and technical fields for producing high-quality, structured documents. Originally developed by Leslie Lamport in the 1980s as a macro package for Donald Knuth's TeX typesetting system [5, 7] LaTeX provides its users with precise control over formatting and is particularly well-suited for documents containing complex mathematical notation, references, and figures. With features like automated citation management, cross-referencing, and modular document structures. It has been considered the standard in disciplines such as mathematics, physics, computer science, and engineering, as extensively documented in works like those of Kottwitz [6] and Mittelbach et al. [9].

Academic journals and conferences often require or strongly encourage LaTeX submissions due to its typographic quality and consistency. Moreover, collaborative platforms like Overleaf [10] and GitHub [2] have further expanded LaTeX's accessibility and integration into the research workflow, allowing teams to co-author manuscripts efficiently using version control systems. In recent years, its role has extended beyond typesetting papers to producing theses, reports, presentations using Beamer [14], and even posters, reinforcing its place in scholarly communication.

Despite its widespread adoption, LaTeX remains fundamentally a markup language that offers considerable freedom in how users structure and format their source files. This flexibility—while empowering—has led to highly divergent authoring styles across individuals and disciplines. As LaTeX continues to evolve alongside increasingly collaborative academic workflows, its role as both a technical and social writing tool becomes more prominent.

3 PROBLEM STATEMENT

LaTex's flexibility in structuring and formatting source files allows users to tailor their documents to specific needs. However, this same freedom also introduces a challenge: there is no broadly accepted coding style or structural convention for LaTeX projects. While programming languages like Java benefit from standardized style guides that improve readability and collaboration [8], LaTeX lacks clear guidelines for organizing source files, naming macros, or structuring preambles. As a result, project structures can vary significantly, leading to confusion, slower onboarding, and difficulties in maintenance and reuse.

Despite LaTeX's central role in scientific communication, little research has explored whether implicit conventions or common practices have emerged across projects. To date, no large-scale analysis has systematically examined LaTeX repositories to determine whether consistent structural patterns exist and whether those patterns might serve as the foundation for a practical, community-informed style guide.

This study addresses that gap by analyzing structural and stylistic practices in publicly available LaTeX repositories. Rather than evaluating the impact of standardization directly, the research seeks to investigate whether a coherent and generalizable coding convention can be derived from how LaTeX is actually used today—and if so, what its key components might be.

TScIT 43, July 4, 2025, Enschede, The Netherlands
© 2025 University of Twente, Faculty of Electrical Engineering, Mathematics and Computer Science.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

3.1 Research Questions

To guide this investigation, the research adopts an exploratory structure. It begins by surveying the diversity of practices across LaTeX projects, then identifies recurring features, and finally considers whether those features could support the creation of a standardized coding convention. This approach is formalized in the following research questions:

- **RQ1:** What structural and stylistic patterns are currently used in LaTeX projects?
- **RQ2:** Which of these features occur frequently and consistently across projects?
- **RQ3:** To what extent could these recurring features form the basis of a standardized LaTeX coding convention?

4 RELATED WORK

Although LaTeX is a cornerstone of scientific and technical writing, few studies have examined its use as a structured coding language. In particular, there is a notable lack of research on how LaTeX source code is organized and whether consistent patterns exist across projects. Insights from adjacent fields, however, provide a strong foundation for exploring this question.

In software engineering, a study by Buse and Weimer [1] demonstrates that consistent coding styles can significantly improve readability and maintainability. These results, while focused on general-purpose programming languages, suggest that even markup languages like LaTeX could benefit from more structured authoring practices, particularly in collaborative settings.

Unlike languages like Python or Java, which offer widely adopted style guides [13, 15], LaTeX users typically develop their own formatting conventions. Foundational texts on LaTeX [9**?**] focus primarily on functionality and typesetting capabilities, offering little guidance on code organization or maintainable project structure.

Complementary research in scientific computing and reproducibility has underscored the importance of documentation, modularity, and consistent practices in computational work [11, 17]. These principles have driven improvements in other technical domains but have not yet been systematically applied to LaTeX, despite its centrality to scholarly communication.

One of the few direct calls for LaTeX standardization is Verna's article *Towards LaTeX Coding Standards* [16], which proposes a set of informal conventions based on programming best practices. Verna highlights the inconsistency of LaTeX source files and argues for clearer structuring, modularization, and naming. However, his proposals are not based on empirical analysis, and no large-scale studies have tested whether LaTeX authors actually follow patterns that could support a shared standard.

This research builds directly on that gap. It seeks to investigate structural and stylistic practices across publicly available LaTeX repositories to assess whether a coherent and generalizable convention can be derived from current usage patterns. In doing so, it extends prior work on code quality and collaboration into the relatively unexplored domain of LaTeX source structure.

5 METHODOLOGY

This research involves the automated collection and analysis of LaTeX source code from publicly available GitHub repositories. To ensure consistency, reproducibility, and scalability, a custom-built tool was developed to handle both data collection and the extraction of features [4].

The LaTeX files were gathered using a Python script specifically created for this study. It systematically searches and filters repositories on GitHub to include only those that meet certain criteria. These inclusion and exclusion rules, which help ensure the relevance and quality of the data, are explained in more detail in the follwing subsection.

Once collected, the projects are passed through another custom tool that extracts useful structural information. The output is saved in a consistent JSON format, which makes it easier to work with later during analysis and visualization.

To find patterns in the data, K-Means clustering was used. This method was chosen for its simplicity, scalability, and effectiveness in partitioning data into distinct groups based on feature similarity. Several cluster sizes—ranging from three to five—were tested, A solution with four clusters provided the best balance between interpretability and granularity, offering enough differentiation to capture meaningful differences without fragmenting the data into overly small groups.

Each cluster is analyzed using a summary table that shows average values for key features, such as number of files, use of macros, or line lengths. These summaries help describe what typical projects in each group look like and are used later to reflect on how LaTeX is actually being used in practice.

The full process—from collecting and filtering the data to analyzing and preparing it—is shown in Figure 1. This structured approach makes it possible to analyze large numbers of LaTeX projects in a consistent way, while making sure that important structural details are preserved and ready for interpretation.

Overall, this automated and scalable setup makes it possible to explore real-world LaTeX usage in depth. It provides a solid foundation for identifying common practices and possible inconsistencies, and supports later discussion about how a more standardized approach to LaTeX coding might look.

5.1 Data Collection

To ensure the relevance and quality of the data, repositories were collected from GitHub using its public API. The search was limited to repositories that included keywords such as `"Thesis"` and `"PhD"`, along with a language filter set to `LaTeX`. These filters were chosen to target projects that likely represent complete academic documents, rather than simple templates or test failes.

To avoid collecting boilerplate content, additional filters were applied. Repositories with keywords like `"template"`, `"example"`, `"sample"`, or `"class"` in their titles or descriptions were excluded. This step helped narrow the focus to projects that reflect real-world usage of LaTeX for academic writing, rather than generic or instructional codebases.

A minimum repository size of 1MB was also enforced. This served as a rough indicator of content richness and helped screen out

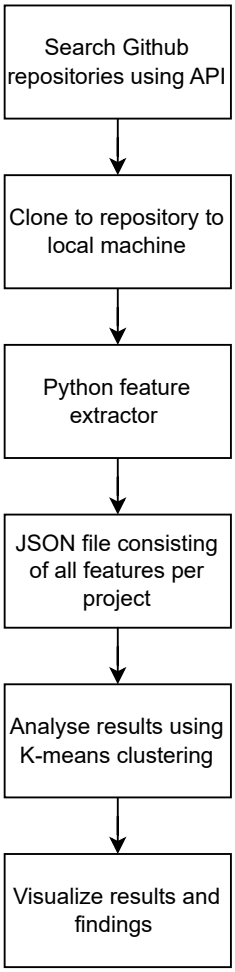

Fig. 1. Flowchart of data collection and analysis

projects that were either incomplete or too small to provide meaningful structural insights.

All selected repositories were cloned locally to make sure the complete folder structure and all associated files were available for processing. Local copies also allowed for manual review in cases where unusual patterns or outliers appeared during analysis. After cloning, each project was passed through an automated feature extraction script, as explained in the next section.

5.2 Dataset

The final dataset as documented in [3], consists of 215 repositories that met all selection criteria. These projects vary in size, structure, and complexity, offering a diverse and well-rounded sample for analysis. Most of the repositories contain academic documents such as theses or dissertations, created by students and researchers from different institutions.

The dataset includes work from a range of academic disciplines and levels, from undergraduate theses to doctoral dissertations. This diversity supports a broad examination of how LaTeX is used in academic contexts and allows for meaningful comparisons across different styles and practices.

5.3 Feature Extraction

The features analyzed in this study were chosen based on both practical considerations and insights from existing literature on software readability, coding style, and document engineering [6, 9]. They were selected to capture dimensions of LaTeX usage that are both measurable and potentially influential in collaborative or maintainable document production.

The features are divided into several thematic categories, each focusing on a different dimension of LaTeX authoring practices. The first category examines the project structure, looking at how repositories are organized on a file-system level. This includes identifying whether a project is composed of a single file or follows a modularized format with multiple included files. Other structural aspects such as the number of subfolders, the use of \input or \include commands, the presence of build or documentation files like Makefile or README.md, and the overall line count are considered as indicators of project complexity and organization.

The second area of features focuses on macro and command usage, assessing how users define and customize commands within their LaTeX documents. This includes counting the number of user-defined macros created using \newcommand or \renewcommand, distinguishing between parameterized and fixed macros, and identifying any redefinitions of standard LaTeX commands. These features reveal how much users rely on LaTeX's extensibility and to what extent they diverge from default behaviors.

Lastly, the study explores readability and coding style, evaluating the source code from a stylistic and formatting perspective. Metrics such as average and maximum line length, indentation consistency (e.g., spaces vs. tabs, and their respective widths), and comment density are analyzed to infer the general clarity and maintainability of the code. These indicators help assess how readable and well-structured LaTeX source files are, especially in collaborative or long-term usage contexts.

By organizing the analysis around these categories—structure, macro usage, and style—the study aims to draw connections between individual authoring practices and broader trends in LaTeX usage. These insights, in turn, support the exploration of whether a standardized LaTeX convention might be feasible or desirable. The detailed results of each category are discussed in the following chapter.

5.4 Clustering and Dimensionality Reduction

To better understand patterns in the extracted features, unsupervised learning techniques were applied to the dataset.

Clustering refers to grouping similar data points together based on the values of their features. In this study, K-Means clustering was used, which partitions the data into a pre-defined number of clusters by minimizing the variance within each cluster. Each repository is assigned to the cluster whose center is nearest in terms of feature distance. This helps reveal latent structures in the data, such as different styles of LaTeX project organization or coding practices.

Because some features are correlated or exist in a high-dimensional space, it can be difficult to visualize or interpret the clusters directly. To address this, *Principal Component Analysis (PCA)* was employed as a dimensionality reduction technique. PCA transforms the original features into a new set of orthogonal variables called principal components, ordered by the amount of variance they capture from the data. By projecting the data onto just the first two or three principal components, it becomes possible to visualize the distribution of repositories and their cluster assignments in a lower-dimensional space while retaining as much of the original information as possible.

Using clustering in combination with PCA thus allows for both quantitative analysis and intuitive visualizations, supporting the interpretation of how LaTeX projects differ across the dataset.

6 RESULTS

This chapter presents the results of the analysis of structural and stylistic practices in LaTeX projects. The findings describe the characteristics observed in the dataset, leaving interpretations and implications for the following discussion chapter.

6.1 Structural Features of LaTeX Projects

The analysis of structural features revealed considerable diversity in how LaTeX projects are organized. Four distinct clusters emerged based on attributes such as the number of `.tex` files, folder structures, total lines of code, and the use of modular commands like \input and \include. These clusters are illustrated in the PCA projection shown in Figure 2, and their distribution across the dataset is summarized in Figure 3. Detailed averages for each cluster appear in Table 1.

Among the clusters, Cluster 0 stands out for its scale and complexity. Projects in this group often comprise dozens of `.tex` files spread across deep folder hierarchies, with total line counts exceeding 13,000 lines. Inclusion commands are consistently present in these repositories.

In contrast, Cluster 2 represents projects on the opposite end of the spectrum. These repositories are comparatively small, averaging fewer than five `.tex` files and around 2,200 lines of code, with shallow folder structures. None of the projects in Cluster 2 use inclusion commands.

Between these two extremes lie Clusters 1 and 3, both of which reflect moderate project scales. Cluster 1 projects typically contain around 21 `.tex` files and about four folders, accompanied by frequent use of inclusion commands. This group also shows a higher prevalence of Makefiles and README files. Cluster 3, while similar in size to Cluster 1, includes slightly fewer folders and projects, and rarely uses Makefiles, although README files remain common.

6.2 Macro and Command Usage

The analysis of macro and command usage in LaTeX projects reports differences in the number of custom macros, the use of parameters, and the redefinition of built-in commands. Clustering of these features resulted in four distinct groups of projects. Figure 4 illustrates the distribution of projects in the feature space, while Figure 5 shows

Fig. 2. Clustering of LaTeX projects based on structural features

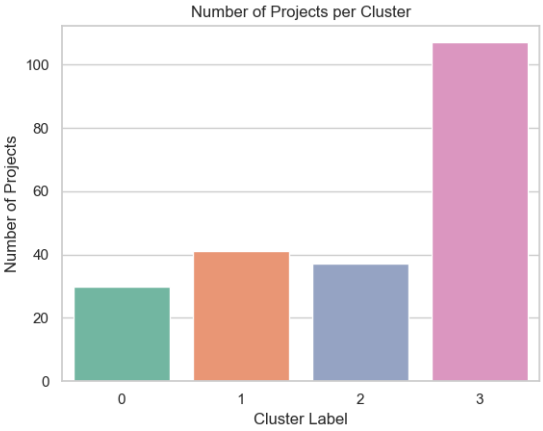

Fig. 3. Number of projects in each cluster based on structural features

Table 1. Average values of structural features in LaTeX projects for each cluster

	files	*folders*	*lines*	*include*[1]	*MakeFile*[1]	*Readme*[1]
0	63.40	15.26	13364.92	1.00	0.13	0.93
1	21.31	4.14	6115.39	0.97	1.00	0.87
2	4.62	1.62	2271.37	0.00	0.29	0.91
3	18.81	3.31	4205.91	1.00	0.00	0.85

Note[1]: Boolean features are represented as floats between 0 and 1, where 1 = true and 0 = false. These include *include*, *MakeFile*, and *Readme*.

the relative size of each cluster. The average values for key features in each cluster are summarized in Table 2.

Cluster 0, the smallest cluster in terms of project count (Figure 5), has an average of 965.75 custom macros per project, all of which use parameters.

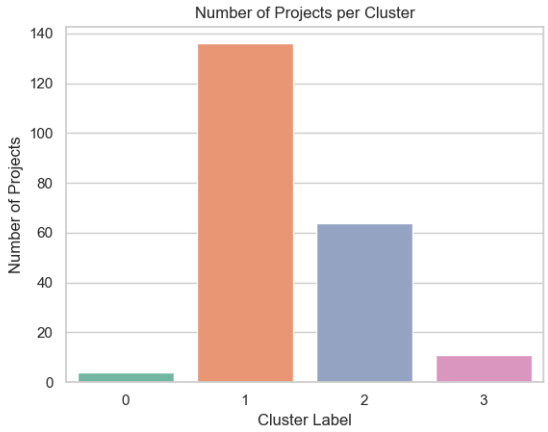

Fig. 4. Clustering of LaTeX projects based on macro and command usage

Fig. 5. Number of projects in each cluster based on macro and command usage

Cluster 1, the largest cluster by project count, has an average of 51.79 custom macros per project, all of which also use parameters. No projects in this cluster redefine built-in LaTeX commands.

Cluster 2 contains projects with an average of 3.59 custom macros per project, none of which use parameters. No projects in this cluster redefine built-in LaTeX commands.

Cluster 3 includes projects with an average of 20.64 custom macros per project. In this cluster, approximately 0.64 custom macros per project use parameters. This cluster is the only one where redefinition of built-in LaTeX commands occurs.

6.3 Readability and Style

The clustering results based on stylistic features show differences in how LaTeX code is formatted across projects. Most repositories in clusters 0, 1, and 3 use space-based indentation. Cluster 2 is the only group that uses tab-based indentation and has shorter average line lengths compared to other clusters.

Table 2. Average values of macro and command usage in LaTeX projects for each cluster

	custom macros	*use parameters*[1]	*redifined buildin*[1]
0	965.75	1.00	0.00
1	51.79	1.00	0.00
2	3.59	0.00	0.00
3	20.64	0.64	1.00

Note[1]*:* Boolean features are represented as floats between 0 and 1, where 1 = true and 0 = false. These include *use parameters*, and *redifined buildin*.

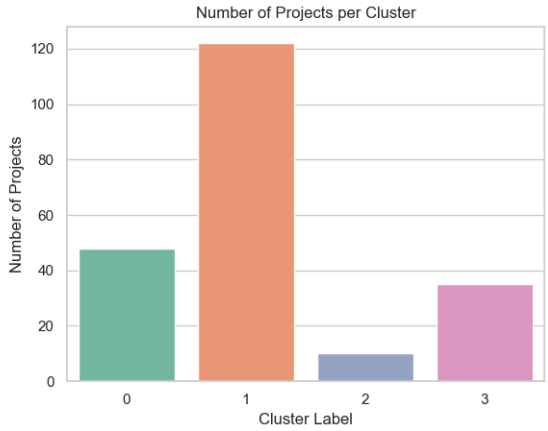

Fig. 6. Clustering of LaTeX projects based on stylistic features

Table 3. Average values of stylistic features in LaTeX projects for each cluster

	comment ratio	*indentation style*[1]	*avg. line*	*longest line*
0	0.06	1.00	104.78	2607.27
1	0.05	1.00	47.88	654.54
2	0.06	0.00	23.82	319.80
3	0.21	1.00	56.70	56.69

Note[1]*:* The *indentation style* feature is represented as a binary value, where 1 = spaces and 0 = tabs.

Cluster 3 has the highest average comment ratio, with comments comprising 21% of all lines. In this cluster, the average line lengths are shorter than in some other groups, and indentation is consistently space-based.

Cluster 0 includes projects with the longest lines observed in the dataset, with some lines exceeding 2,600 characters. The average comment ratio in cluster 0 is moderate compared to the other clusters.

Projects in cluster 1 show moderate average line lengths and a lower comment ratio relative to cluster 3.

Table 3 summarizes the average values for comment ratio, indentation style, and line lengths for each cluster.

Fig. 7. Number of projects in each cluster based on stylistic features

7 DISCUSSION AND CONCLUSION

This study set out to examine how LaTeX is used across a diverse set of real-world academic projects. The clustering analyses revealed notable variation in both structural and stylistic practices, suggesting that while LaTeX is a powerful tool, its flexibility also leads to fragmented usage patterns without consistent conventions.

7.1 Structural Patterns

The structural analysis uncovered a continuum ranging from highly modular projects (Cluster 0) to minimalistic, single-file projects (Cluster 2). The projects in Cluster 0 demonstrate practices akin to software engineering principles: extensive use of inclusion commands, numerous files, and deep folder hierarchies. Such organization is likely beneficial for large-scale documents like PhD theses or collaborative writing efforts where chapters, figures, and appendices are managed as separate entities. However, the relatively low presence of Makefiles even in this cluster suggests that automation tools are not yet universally adopted, possibly because academic authors may prioritize content over tooling or may lack familiarity with build automation.

Clusters 1 and 3 represent a middle ground where projects are moderately modular but with simpler folder structures and fewer files. Cluster 1 distinguishes itself by widespread use of Makefiles, indicating an inclination toward automated compilation and workflow efficiency. In contrast, Cluster 3, although similarly modular, largely omits automation tools. This divergence might reflect different author profiles—some prioritizing reproducibility and automation, others focusing on lightweight setups.

Cluster 2, characterized by single-file projects with few lines of code, indicates a minimalist approach. Such projects may correspond to shorter documents like reports or coursework. The absence of inclusion commands suggests authors of these projects either lack awareness of modular practices or deem them unnecessary for small documents. While simpler to maintain for short texts, this approach may become unwieldy as documents grow in complexity.

Collectively, these patterns underscore the lack of standardized practices for organizing LaTeX projects. Despite LaTeX's maturity, there appears to be no widespread consensus on best practices for file structuring, inclusion commands, or automation tooling. This variability could pose challenges for maintainability, collaboration, and onboarding of new contributors.

7.2 Macro and Command Usage Patterns

The clustering of macro and command usage revealed significant differences in how users extend LaTeX's functionality. Cluster 0, although a small outlier, showcases an extreme level of customization with nearly 1,000 custom macros per project. Such heavy macro use may indicate specialized document classes, automated document generation, or very advanced users. However, the rarity of these projects suggests that this level of customization is not representative of general practice.

Cluster 1, the largest group, balances moderate custom macro use with widespread parameterization. This approach indicates a practical use of LaTeX's extensibility to simplify repetitive tasks without altering core LaTeX behavior. It's a sign of users seeking efficiency and consistency in document preparation while adhering to LaTeX's standard conventions.

Clusters 2 and 3 differ significantly. Cluster 2's minimal macro usage reflects reliance on LaTeX's built-in commands and simpler documents, echoing the structural minimalism observed earlier. Conversely, Cluster 3 shows moderate macro use and is unique in redefining built-in commands, suggesting a more experimental approach. Authors in Cluster 3 may be exploring custom document classes or adapting LaTeX for specialized outputs, possibly for niche use cases or specific institutional requirements.

The divergence in macro usage illustrates how LaTeX's flexibility can result in both simple, default usage and highly customized environments.

7.3 Readability and Style Patterns

Stylistic differences were also evident. Cluster 3 projects stand out for their high comment ratios, suggesting an emphasis on documenting code for clarity or collaboration. This practice could be linked to collaborative projects where readability and maintainability are critical, or to pedagogical contexts where code serves instructional purposes.

Cluster 2 projects, distinguished by tab-based indentation and shorter line lengths, may reflect authors using different editors, toolchains, or conventions. While tabs can offer flexibility, their inconsistent rendering across environments may reduce readability. Meanwhile, extremely long lines in Cluster 0 suggest either auto-generated content or dense macro definitions, potentially hampering readability and maintainability.

The variation in code style across clusters highlights the absence of widely adopted LaTeX style guidelines. Unlike programming languages such as Python, which enforce style through tools like PEP8, LaTeX lacks a unified standard for formatting or code readability. This gap can hinder collaboration and onboarding, particularly in multi-author projects.

7.4 Addressing the Research Questions

In relation to *RQ1: What structural and stylistic patterns are currently used in LaTeX projects?*, the results show significant diversity. Structurally, projects range from highly modular, multi-file designs with complex folder hierarchies (Cluster 0) to minimalistic, single-file documents (Cluster 2). Clusters 1 and 3 fall in between, with moderate modularity and simpler folder structures. Stylistically, projects vary in comment density, line lengths, and indentation styles. For example, Cluster 3 shows a high level of inline documentation, while Cluster 2 is characterized by tab-based indentation and shorter lines.

Regarding macro and command usage, considerable variation was observed as well. Cluster 0 projects exhibit exceptionally high numbers of custom macros—often with parameterization—suggesting highly tailored document setups. Cluster 1, while less extreme, still features consistent use of parameterized custom macros. In contrast, Cluster 2 shows minimal reliance on custom macros, using LaTeX largely in its default form, while Cluster 3 shows moderate macro use and is the only cluster where redefinitions of built-in commands were detected.

In relation to *RQ2: Which of these features occur frequently and consistently across projects?*, several recurring practices emerged. Common structural features include the use of `\input` and `\include` commands for modular document structuring, the presence of README files, and a preference for space-based indentation. In macro usage, many projects define custom macros, but the majority avoid redefining built-in commands, indicating a balance between customization and maintaining LaTeX's standard behavior. These patterns appear most consistently in Clusters 1 and 3.

7.5 Toward a Convention

In addressing *RQ3: To what extent could these recurring features form the basis of a standardized LaTeX coding convention?*, the results indicate that while no universal standard currently governs LaTeX project organization or code style, certain informal conventions are emerging across many projects. Practices such as structuring documents into multiple files using \input or \include, maintaining a README file for documentation, using space-based indentation, and defining custom macros without extensively redefining built-in commands appear frequently and consistently in the analyzed repositories.

These recurring patterns suggest that a flexible, community-driven style guide could be developed, reflecting common practices already in use. Rather than imposing rigid standards, such a guide could help improve clarity, support collaboration, and simplify maintenance—particularly in academic and multi-author contexts—while still allowing users to adapt their workflows to different project sizes and disciplines.

7.6 Limitations and Future Work

This study is not without limitations. The dataset was restricted to GitHub repositories matching certain keywords, potentially overlooking other relevant LaTeX practices. Additionally, the clustering analysis, while informative, abstracts over contextual details such as discipline-specific needs or individual author preferences.

Future research could expand the dataset, explore additional metadata (e.g., academic domain or institutional origin), or engage users in evaluating proposed conventions. Furthermore, it would be valuable to examine the impact of adopting such conventions on document quality, maintainability, or collaboration efficiency.

7.7 Closing Remarks

Ultimately, this research offers a first step toward grounding LaTeX conventions in empirical usage patterns. By making visible the diverse—but often repeated—ways in which LaTeX is used in the wild, it opens the door to creating shared practices that support better documentation, smoother collaboration, and more maintainable academic writing.

REFERENCES

[1] Raymond P.L. Buse and Westley Weimer. 2010. Learning a metric for code readability. *IEEE Transactions on Software Engineering* 36, 4 (2010), 546–558. https://doi.org/10.1109/TSE.2009.70

[2] GitHub, Inc. 2024. GitHub. https://github.com.

[3] Bart Griepsma. 2025. LaTeX Academic Dataset. https://github.com/Bart0TW/LaTeX_academic_dataset. Accessed: 2025-06-22.

[4] Bart Griepsma. 2025. LaTeX Project Analyzer. https://github.com/Bart0TW/LaTeX-Project-Analyzer. Accessed: 2025-06-27.

[5] Donald E. Knuth. 1984. *The TeXbook*. Addison-Wesley.

[6] Stefan Kottwitz. 2011. *LaTeX Beginner's Guide*. Packt Publishing.

[7] Leslie Lamport. 1994. *LaTeX: A Document Preparation System* (2nd ed.). Addison-Wesley.

[8] T. Lee, J. B. Lee, and H. P. In. 2013. A study of different coding styles affecting code readability. *International Journal of Software Engineering and Its Applications* 7, 2 (2013). https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=17274fef400424fe4876cd23edb8318e4944b203

[9] Frank Mittelbach, Michel Goossens, Johannes Braams, David Carlisle, and Chris Rowley. 2021. *The LaTeX Companion* (3rd ed.). Addison-Wesley Professional.

[10] Overleaf Ltd. 2024. Overleaf: Online Collaborative LaTeX Editor. https://www.overleaf.com.

[11] João Felipe Pimentel, Leonardo Murta, Vanessa Braganholo, and Juliana Freire. 2019. A Large-Scale Study About Quality and Reproducibility of Jupyter Notebooks. In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. 507–517. https://doi.org/10.1109/MSR.2019.00077

[12] Margaret-Anne Storey, Leif Singer, Brendan Cleary, Fernando Figueira Filho, and Ariel Zagalsky. 2014. The (r)evolution of social coding: GitHub as a collaborative social network. In *Proceedings of the Future of Software Engineering*. ACM, 100–103.

[13] Sun Microsystems. 1999. Code Conventions for the Java Programming Language. https://www.oracle.com/java/technologies/javase/codeconventions-contents.html. Accessed: 2024-06-28.

[14] Till Tantau, Joseph Wright, and Vedran Miletić. 2023. *The Beamer Class: User Guide*. LaTeX Project. Version 3.70.

[15] Guido van Rossum, Barry Warsaw, and Nick Coghlan. 2001. PEP 8 – Style Guide for Python Code. https://peps.python.org/pep-0008/. Accessed: 2024-06-28.

[16] Didier Verna. 2011. Towards LaTeX Coding Standards. *TUGboat* 32, 3 (2011), 309–315. https://www.tug.org/TUGboat/tb32-3/tb102verna.pdf

[17] Greg Wilson, DA Aruliah, CT Brown, NP Chue Hong, M Davis, RT Guy, SHD Haddock, KD Huff, IM Mitchell, MD Plumbley, et al. 2014. Best practices for scientific computing. *PLOS Biology* 12, 1 (2014), e1001745. https://doi.org/10.1371/journal.pbio.1001745