CODE CONTRIBUTION AND AUTHORSHIP

Eva Maxfield Brown

University of Washington Information School

evamxb@uw.edu

Isaac Slaughter

University of Washington Information School

Shahan Ali Memon

University of Washington Information School

Nicholas Weber

University of Washington Information School

nmweber@uw.edu

February 1, 2025

ABSTRACT

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur eget porta erat. Morbi consectetur est vel gravida pretium. Suspendisse ut dui eu ante cursus gravida non sed sem. Nullam sapien tellus, commodo id velit id, eleifend volutpat quam. Phasellus mauris velit, dapibus finibus elementum vel, pulvinar non tellus. Nunc pellentesque pretium diam, quis maximus dolor faucibus id. Nunc convallis sodales ante, ut ullamcorper est egestas vitae. Nam sit amet enim ultrices, ultrices elit pulvinar, volutpat risus.

1 Introduction

Contemporary scientific research fundamentally depends on specialized software tools and computational methods (Edwards et al. 2013; Mayernik et al. 2017; Howison et al. 2015). While these tools - from analysis scripts for data processing to infrastructure for data collection (Hasselbring et al. 2024) - have become essential to modern research practices, their development and maintenance face persistent challenges in receiving appropriate academic recognition (Muna et al. 2016). This tension is particularly acute given software's expanding roles: enabling reproducible research and large-scale experiments (Krafczyk et al. 2019; Trisovic et al. 2021), serving as a detailed log of research methodology (Ram 2013), and increasingly being shared alongside research articles as a crucial research artifact (Cao et al. 2023; Trujillo, Hébert-Dufresne, and Bagrow 2022).

This misalignment between software's importance and its recognition has significant consequences for academic careers. Software contributions are often relegated to acknowledgments rather than warranting authorship (Philippe et al. 2019), and this lack of formal credit can significantly impact career advancement in academia (Carver et al. 2022; Biagioli and Galison 2014). The academic community continues to grapple with developing appropriate systems for software citation and credit attribution that adequately reflect these contributions (Merow et al. 2023; Westner et al. 2024; Katz et al. 2020).

Recent initiatives like the Contributor Roles Taxonomy (CRediT) have attempted to address these challenges by expanding academic authorship criteria to include specialized roles (Brand et al. 2015), but have not fully resolved the issues surrounding software contribution recognition. While previous research has used CRediT and similar systems to understand research labor distribution (Larivière, Pontille, and Sugimoto 2020; Larivière et al. 2016; Sauermann and Haeussler 2017; K. Li, Zhang, and Larivière 2023; Lu et al. 2019), these frameworks remain largely centered on traditional author lists. This traditional focus perpetuates existing problems, including historical and systematic bias, and relies heavily on self-reporting without external verification (Haeussler and Sauermann 2013; Gøtzsche et al. 2007; Ni et al. 2021). The limitations of current contribution frameworks, combined with the growing importance of software in research, highlight the need for a systematic examination of how code contributions actually relate to academic recognition and impact.

These challenges raise several critical questions about the relationship between software development and academic recognition:

- How does the distribution of coding contributions within research teams relate to article impact?
- How do traditional academic roles align with actual code contributions?
- How are code-contribution patterns associated with an individual's scientific impact metrics?

The emergence of public code repositories alongside published research creates a unique opportunity to move beyond self-reported contribution data. Version control systems maintain detailed records of who contributes what code and when, providing an unprecedented window into the actual patterns of software development in scientific research. However, leveraging this data requires solving significant technical challenges in connecting repository information with traditional academic publishing records.

To address these questions, we developed a novel predictive model that enables systematic matching between scientific article authors and source code developer accounts. This approach was necessary due to the lack of standardized identifiers (like ORCID) for developers (Haak et al. 2012) and inconsistencies in naming and email conventions across platforms. We leverage recent advances in transformer-based architectures and semantic embeddings to handle subtle variations in identity information (Y. Li et al. 2020; Brunner and Stockinger 2020).

By applying our model across a corpus of 138596 paired research articles and repositories, we uncover several striking patterns in the relationship between code contributions and academic recognition. Our analysis of source-code repositories contribution histories reveals that approximately 25% of articles have non-author code-contributors. We find that code-contributing authors are associated with modest increases in article-level impact metrics (~4.5% increase in citations per code-contributing author), though these effects become statistically non-significant when controlling for domain, article type, and open access status. First authors emerge as significantly more likely to be code contributors compared to other positions across all conditions tested. And most notably, we find a clear negative relationship between coding frequency and scholarly impact - authors who contribute code more frequently show progressively lower h-indices compared to their non-coding peers, a pattern that persists across various controls for an author's number of total works, author position, domain, and article type.

These findings reveal fundamental tensions between software development and traditional academic recognition systems. While our results demonstrate the prevalence of software contributions in modern research (both credited and un-credited), they also suggest that current academic metrics may systematically undervalue technical contributions. This quantitative evidence provides an empirical foundation for ongoing discussions about academic credit systems and raises important questions about how institutions can better align recognition with the full spectrum of contributions that drive scientific progress.

2 Background

The relationship between scientific software development and academic credit systems represents a complex intersection of traditional academic practices and modern research requirements. While academic credit has historically focused on analytical, theoretical, and experimental contributions (Larivière et al. 2016; Liu, Zhang, and Li 2023), software development has long been viewed as purely technical rather than scholarly work. However, there is growing recognition that research software development requires deep domain expertise (Heroux 2022; Carver et al. 2022), particularly as increased emphasis on large-scale data projects has amplified the need for sophisticated computational methods (Jin et al. 2015; Hampton et al. 2013; Fan, Han, and Liu 2014).

Understanding the dynamics of scientific software development requires examining both team-level patterns and individual contributions. Previous research has extensively studied how team composition affects research outcomes, finding that larger and more diverse teams often produce higher-impact work (Franceschet and Costantini 2010; Larivière et al. 2014; AlShebli et al. 2024). Similarly, software engineering studies have demonstrated that larger development teams typically produce higher quality code (Wyss, De Carli, and Davidson 2023; Meirelles et al. 2010). However, we lack systematic evidence of how these patterns translate to academic software development, where contribution recognition follows different rules than traditional software projects. Given that technically sophisticated research projects often enable novel scientific insights and methodological advances, **we hypothesize that projects with more code contributors will receive higher citation counts, reflecting the value of technical contributions even when they go uncredited (H1)**.

Academic authorship conventions establish clear expectations about intellectual contribution and project responsibilities, though these vary by field and institution (Larivière et al. 2016; Larivière, Pontille, and Sugimoto 2020). First authors typically bear primary responsibility for intellectual and experimental contributions, while corresponding authors maintain research artifacts and serve as primary points of contact (Júnior et al. 2016). The increasing importance of computational methods suggests that many first authors, who are commonly also an article's corresponding author, may need to directly engage with software development to execute their research vision (Heroux 2022). **Given these expectations, we hypothesize that first authors (H3) and corresponding authors (H4) will show higher rates of code contribution compared to middle, last, and non-corresponding authors**.

The tension between software development and academic advancement reflects fundamental misalignments in how different types of contributions are valued and measured. Traditional impact metrics were designed to capture theoretical and experimental advances (Biagioli and Galison 2014), while software contributions often manifest through different channels. **We hypothesize that researchers who frequently contribute code will be associated with lower individual-level impact metrics compared to their non-coding peers (H5)**. This pattern likely emerges through multiple mechanisms:

1. Lack of standardized software citation practices reduces the formal recognition of technical contributions
2. Theoretical advances may garner more citations than methodological improvements
3. Version-specific iterative software publications split citations across multiple articles

These mechanisms help explain the career challenges faced by researchers who prioritize technical contributions, particularly in fields where software development is not yet recognized as a core scholarly activity.

3 Data and Methods

3.1 Linking Scientific Articles and Source Code Repositories

Modern scientific research increasingly requires the public sharing of research code, creating unique opportunities to study the relationship between academic authorship and software development. Many journals and platforms now require or recommend code and data sharing (Stodden, Guo, and Ma 2013; Sharma et al. 2024), creating traceable links between publications and code that enable systematic study of both article-repository and author-developer relationships (Hata et al. 2021; Kelley and Garijo 2021; Stankovski and Garijo 2024; Milewicz, Pinto, and Rodeghero 2019).

Our data collection process leverages multiple sources of linked scientific articles and code repositories to ensure broad coverage of multiple different domains and article types. Our dataset combines article-source-code-repository pairs from:

- PLOS: Traditional research articles
- JOSS and SoftwareX: Specialized software-focused publications
- Papers with Code (ArXiv): preprints

To reduce the complexity of dataset processing and enrichment, we filter out any article-source-code-repository pairs which store code somewhere other than GitHub. While this decision prioritizes processing simplicity, we acknowledge that while GitHub is the predominate host of scientific software, it is also stored and shared on other platforms, which should be investigated as a part of future research (Trujillo, Hébert-Dufresne, and Bagrow 2022).

Through integration of multiple data sources, we extract detailed information about both the academic and software development aspects of each project:

Through integration of multiple data sources, we extract detailed information about both the academic and software development aspects of each project. We utilize the Semantic Scholar API for DOI resolution to ensure that we find the latest version of each article, which is particularly important for working with preprints as they may have been published in a journal since their inclusion in the Papers with Code dataset. We then utilize the OpenAlex API to gather publication metadata (open access status, domain, publication date), author details (name, author position, corresponding author status), and article- and individual-level metrics (citation counts, FWCI, h-index). The GitHub API provides similar information for source code repositories, including repository metadata (name, description, languages, creation date), contributor details (username, name, email), and repository-level metrics (star count, fork count, issue count).

Taken together, we form one of the largest and most comprehensively annotated collections of paired scientific articles and associated source-code repositories. In total, we collect and enrich data for 163292 article-repository pairs.

3.2 A Predictive Model for Matching Article Authors and Source Code Contributors

3.2.1 Annotated Dataset Creation

The development of an accurate author-developer matching model requires high-quality labeled training data that captures the complexity of real-world identity matching. Entity matching between authors and developers is non-trivial due to multiple forms of name variation and incomplete information. These variations can include differences in formatting (e.g., “J. Doe” vs “Jane Doe”), institutional versus personal email addresses, and incomplete or outdated information.

We developed an annotation process to create a robust training dataset while maximizing efficiency and accuracy. We focused our annotation efforts on JOSS articles to increase positive match density, as these software-focused publications typically have higher overlap between authors and developers. For each JOSS author, we generated three random pairings with developer accounts from the article’s associated repository. From the full set of generated pairs, we randomly sampled 3,000 for annotation which two independent annotators then labeled as either a match or non-match. After completing all annotations, we systematically resolved any disagreements between the annotators through discussion and additional verification.

The resulting annotated dataset provides a comprehensive foundation for training our predictive model while highlighting common patterns in author-developer identity matching. After resolving all annotated pairs, our final dataset contains:

1. Match Distribution:
   - 451 (15.0%) positive matches
   - 2548 (85.0%) negative matches
2. Unique Individuals:
   - 2027 unique authors
   - 2733 unique developer accounts
3. Developer Profile Completeness:
   - 2191 (80.2%) accounts have associated names
   - 839 (30.7%) accounts have associated emails

3.2.2 Training and Evaluation

Our training and evaluation methodology begins with careful dataset preparation to prevent data leakage between training and test sets. To ensure complete separation of both authors and developers, we randomly selected 10% of unique authors and 10% of unique developers, designating any pairs containing these selected entities for the test set. Due to the combinatorial nature of our author-developer pairs, this entity-based splitting strategy resulted in 2442 (81.4%) pairs for training and 557 (18.6%) pairs for testing.

For our predictive model, we evaluate three transformer-based architectures that have demonstrated strong performance in entity matching tasks: DeBERTa-v3-base (He, Gao, and Chen 2021; He et al. 2021), mBERT (bert-base-multilingual-cased) (Devlin et al. 2018), and DistilBERT (Sanh et al. 2019). While BERT-based architectures have been widely studied, they continue to achieve state-of-the-art results across various natural language processing tasks, particularly in scenarios requiring precise entity matching and relationship identification (Tran et al. 2024; Yu et al. 2024; Jeong and Kim 2022).

We conducted systematic evaluation of these base models across different combinations of developer-account features, ranging from using only the username to incorporating full profile information (username, display name, and email address). All models were fine-tuned using the Adam optimizer with a learning rate of 1e-05, batch sizes of 8 for both training and evaluation, and a linear learning rate scheduler. Given the size of our dataset and the binary nature of our classification task, models were trained for a single epoch to prevent overfitting.

Model performance was assessed using standard binary classification metrics, with particular emphasis on the F1 score for positive (matching) pairs due to the inherent class imbalance in author-developer matching. This evaluation framework allows us to directly compare model architectures and feature combinations while accounting for both precision and recall in identifying correct matches.

Our comprehensive model evaluation revealed that fine-tuning DeBERTa-v3-base (He, Gao, and Chen 2021) with developer username and display name as input features produces optimal performance for author-developer matching. This model configuration achieves a binary F1 score of 0.944, with accuracy of 0.984, precision of 0.938, and recall of 0.95. A complete comparison of model architectures and feature combinations can be found in Table 4.

Analysis of each model's performance revealed that including developer display names has the largest positive impact on model performance compared to username alone. While this effect might be partially attributed to the higher availability of display names in our dataset compared to email addresses, the performance improvement is notable. We additionally observe that mBERT's performance was comparable with DeBERTa's while additionally using the developer email address as an additional input feature, but selected the DeBERTa configuration for its relative simplicity and more recent and comprehensive pretraining. DeBERTa's consistent strong performance across various feature combinations, combined with its more extensive pretraining dataset, suggests better generalization potential for future applications.

To facilitate the reuse of our work, we have made our trained model and supporting code publicly available. Complete fine-tuning, evaluation, and inference code is available as the Python package: sci-soft-models, and the fine-tuned model has been deployed to HuggingFace (evamxb/dev-author-em-clf).

3.3 Linking Authors and GitHub Developer Accounts

Our trained entity-matching model enables comprehensive identification of author-developer relationships across all possible author and developer-account combinations within each article-repository pair. This broad application accounts for the complex realities of academic software development practices, particularly the common occurrence of researchers maintaining multiple developer accounts across different projects or institutions, and account transitions as researchers move between roles.

While our model demonstrates strong performance, we acknowledge certain limitations in our approach. Notably, the model's performance can be affected by shorter names (both usernames and display names) where less textual information is available for matching. Additionally, while organization accounts (such as research lab accounts used for project management) present a potential challenge for accurate matching, our filtering mechanisms applied before analysis help minimize their impact in modeling.

The resulting dataset represents, to our knowledge, the first large-scale collection of linked article-repository and author-developer-account pairs, particularly in the physical sciences. Specifically, our dataset contains 138596 article-repository pairs, with 295806 distinct authors and 152170 distinct developer accounts. From these distinct entities, we identify 108754 annotated author-developer pairs. A detailed breakdown of these counts by data source, domain, document type, and open access status is available in Table 1.

Table 1: Counts of Article-Repository Pairs, Authors, and Developers by Data Sources, Domains, Document Types, and Access Status.

Category	Subset	Article-Repository Pairs	Authors	Developers
By Domain	Physical Sciences	116600	240545	130592
	Social Sciences	8838	29269	14043
	Life Sciences	7729	31649	12150
	Health Sciences	5172	25979	7248
By Document Type	preprint	72177	170301	87311
	research article	63528	173183	78935
	software article	2891	9294	12868
By Access Status	Open	132856	286874	147831
	Closed	5740	23668	9352
By Data Source	pwc	129615	262889	134926
	plos	6090	30233	8784
	joss	2336	7105	11362
	softwarex	555	2244	1628
Total		**138596**	**295806**	**152170**

4 Preliminary Analysis Code Contributor Authorship and Development Dynamics of Research Teams

4.1 Software Development Dynamics Within Research Teams

Understanding the composition and dynamics of software development teams provides essential context for analyzing how code contributions relate to academic recognition and impact. To ensure reliable analysis, we focus on a subset of our data that includes only article-repository pairs with at least one article citation, repository commit activity that at the latest falls within 90 days of publication, and research teams of typical size (removing those with fewer than 3 authors or above the 97th percentile of team sizes).

Within this filtered dataset, we categorized individuals into three groups: code-contributing authors (CC-A) who both authored papers and contributed code to associated repositories, non-code-contributing authors (NCC-A) who authored papers but showed no evidence of code contributions, and code-contributing non-authors (CC-NA) who contributed code but received no authorship recognition. This categorization revealed that papers in our dataset typically have 4.9 ś 1.9 total authors, with 1.0 ś 0.7 code-contributing authors and 3.9 ś 2.0 non-code-contributing authors. Beyond the author list, papers averaged 0.4 ś 1.7 code-contributing non-authors. Table 2 provides a detailed breakdown of these distributions by domain, article type, and open access status.

Perhaps most striking is our finding that 6237 papers (27.4%) have at least one code contributor who did not receive authorship recognition. Within this substantial subset of papers, we found an average of 1.6 ś 3.0 unrecognized code contributors per paper. The presence of only one code-contributing author per paper, on average, aligns with previous research by Larivière, Pontille, and Sugimoto (2020) showing that technical tasks like data curation, formal analysis, visualization, and software development typically fall to first authors. However, our finding that over a quarter of papers have unrecognized code contributors suggests a more complex dynamic between software development and authorship recognition.

Table 2: Mean and Standard Deviation of Non-Code-Contributing Authors (NCC-A), Code-Contributing Authors (CC-A), and Code-Contributing Non-Authors (CC-NA) Research Team Members by Domain, Article Type, and Open Access Status. Only includes research teams from article-repository pairs with a most recent commit no later than 90 days after publication and excludes research teams which are in the top 3% of total author sizes.

Control	**Subset**	**Total Authors**	**NCC-A**	**CC-A**	**CC-NA**
OA Status	Closed	5.1 ś 1.9	4.0 ś 1.9	1.1 ś 0.7	0.5 ś 2.1
	Open	4.9 ś 1.9	3.9 ś 2.0	1.0 ś 0.7	0.4 ś 1.7
Domain	Health Sciences	6.1 ś 2.5	5.1 ś 2.5	1.0 ś 0.6	0.4 ś 1.2
	Life Sciences	5.2 ś 2.1	4.2 ś 2.2	1.0 ś 0.7	0.4 ś 1.2
	Physical Sciences	4.8 ś 1.8	3.8 ś 1.9	1.0 ś 0.7	0.5 ś 1.8
	Social Sciences	4.5 ś 1.7	3.5 ś 1.8	1.1 ś 0.7	0.3 ś 1.1
Article Type	preprint	4.8 ś 1.8	3.8 ś 1.9	1.1 ś 0.7	0.5 ś 2.2
	research article	4.9 ś 1.9	3.9 ś 2.0	1.0 ś 0.7	0.4 ś 1.6
	software article	4.7 ś 1.9	3.2 ś 1.9	1.5 ś 1.4	0.9 ś 1.1
Overall		**4.9 ś 1.9**	**3.9 ś 2.0**	**1.0 ś 0.7**	**0.4 ś 1.7**

When examining these patterns over time and across different team sizes (Figure 1), we found that both the number of code-contributing authors and unrecognized contributors has remained relatively stable. This stability suggests that while the exclusion of code contributors from authorship isn't worsening, it represents a persistent feature of academic software development rather than a historical artifact or transition period in academic practices.

4.1.1 Modeling Article Citations

Building upon previous work examining the effects of team size and team diversity on scientific impact and software quality (see Section 2), we investigate how the number of code contributors within a research team may be associated with an article's research impact. We hypothesize that more code contributors may signal greater technical complexity in research, which may be associated with higher citation counts as the community builds upon more technically sophisticated works.

Using our filtered dataset of article-repository pairs, we conducted multiple regression analyses to examine these relationships while controlling for various factors. Without controlling for domain, open access, or article type differences (Table 5), we found a modest positive association between the number of code-contributing authors and article citations, with each code-contributing author being associated with a 4.5% increase in article citations ($p < 0.001$).

When controlling for article type (Table 8), we observed divergent patterns between preprints and research articles. For preprints, each code-contributing non-author was associated with a statistically significant 2.9% decrease in citations ($p < 0.01$). In contrast, research articles showed more positive associations: we found a significant positive relationship between code-contributing authors and citations ($p < 0.001$), though we cannot estimate the precise magnitude due to the non-significant main effect in the model. Additionally, each code-contributing non-author was associated with a 0.8% increase in expected citations for research articles ($p < 0.001$).

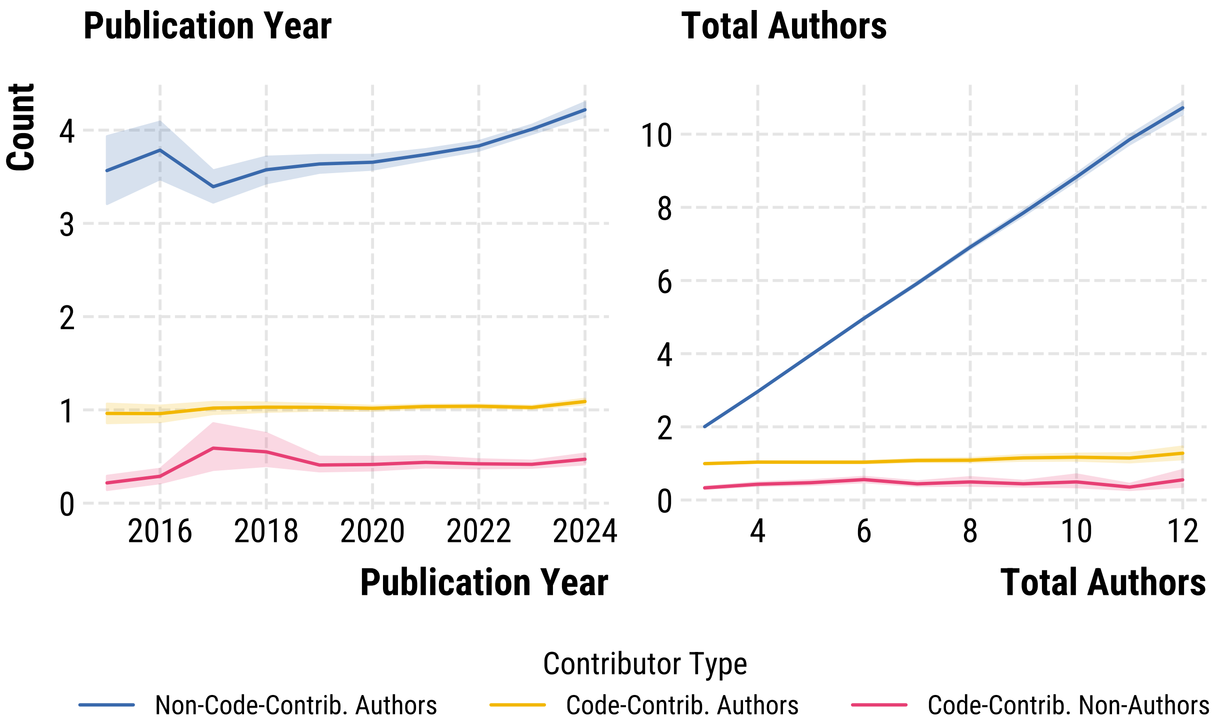

Figure 1: Mean number of Non-Code-Contributing Authors, Code-Contributing Authors, and Code-Contributing Non-Authors by Publication Year and by Total Number of Authors. Only includes article-repository pairs with a most recent commit no later than 90 days after publication and excludes research-teams which are in the top 3% of total author sizes for publication years with 50 or more articles.

Overall, while we find statistically significant associations between code contributions and citation counts, these effects are relatively modest in magnitude.

4.2 Characteristics of Scientific Code Contributors

4.2.1 Author Positions of Code Contributing Authors

Building upon previous work examining the relationship between authorship position and research contributions, we investigate how author position may relate to code contribution patterns. We hypothesize that first authors, who traditionally contribute the bulk of intellectual and experimental work, would be most likely to contribute code to a project, while middle and last authors, who often provide oversight and guidance, would be less likely to contribute code.

To analyze these patterns within our filtered dataset of article-repository pairs (excluding those with most recent commit dates more than 90 days after publication), we first conducted chi-square tests of independence between author position and code contribution status. These tests revealed significant associations both overall and when controlling for research domain, article type, and open access status (all $p < 0.01$). Following these significant results, detailed post-hoc binomial tests (Table 9) revealed clear position-based differences: 68.6% of first authors contributed code to their projects, compared to only 9.0% of middle authors and 9.1% of last authors. These differences remained statistically significant across all tested scenarios, regardless of research domain, article type, or open access status.

These patterns strongly align with traditional academic authorship conventions, where first authors typically take primary responsibility for both intellectual and technical aspects of the research, while middle and last authors more commonly provide oversight and guidance. The consistency of these findings across different subsets of our data suggests a deeply embedded relationship between author position and technical contributions in academic software development.

4.2.2 Corresponding Status of Code Contributing Authors

Building upon our analysis of author position, we next examine how corresponding author status relates to code contribution patterns. We hypothesize that corresponding authors, who traditionally maintain research artifacts and serve as primary points of contact, would be more likely to contribute code compared to non-corresponding authors, as this role often involves responsibility for project resources and materials.

To analyze these relationships within our filtered dataset of article-repository pairs, we conducted chi-square tests of independence between corresponding author status and code contribution status. Surprisingly, these tests revealed patterns contrary to our initial hypothesis. Both corresponding and non-corresponding authors were significantly less likely to be code contributors than would be expected by chance (p < 0.01), with only 30.2% of corresponding authors and 20.5% of non-corresponding authors contributing code. Detailed post-hoc binomial tests (Table 10) revealed this pattern holds true across nearly all conditions, with only two notable exceptions: corresponding authors in software-focused articles and those in closed-access publications showed no significant difference in their likelihood to contribute code.

These findings challenge conventional assumptions about the relationship between corresponding authorship and technical contributions. While corresponding authors are traditionally responsible for maintaining research artifacts, our results suggest this responsibility may not typically extend to direct engagement with software development.

4.2.3 Modeling Author H-Index

Table 3: Counts of Researcher Coding Status Used in H5

Control	Subset	Any Coding	Majority Coding	Always Coding	Total
Freq. Author Pos.	First	1689	4952	3322	11444
	Middle	12184	3327	701	31911
	Last	2413	568	191	10188
Freq. Domain	Health Sciences	345	202	86	1501
	Life Sciences	369	241	129	1436
	Physical Sciences	15289	8179	3821	49311
	Social Sciences	283	225	178	1295
Freq. Article Type	Preprint	9572	4963	2214	28948
	Research Article	6669	3765	1871	24217
	Software Article	45	119	129	378

Building upon previous work examining career implications for researchers who prioritize software development Section 2, we investigated how varying levels of code contribution relate to scholarly impact through h-index metrics. To ensure a robust analysis, we applied several key data filtering steps. We only included researchers with at least three publications in our dataset, removed those with more than three developer account associations, and used each researcher's most common (or most recent) domain, article type, and author position, with ties broken by the most recent occurrence. We also removed h-index outliers by excluding researchers below the bottom 3rd and above the top 97th percentiles.

We categorized researchers' coding contributions into mutually exclusive groups: non-coders (no code contributions), any coding (code contribution in less than half of article-repository pairs), majority coding (code contribution in at least half, but not all, article-repository pairs), and always coding (code contribution in every article-repository pair).

Our analysis revealed a consistent and statistically significant negative relationship between code contribution frequency and h-index across multiple analytical controls. In our initial uncontrolled analysis (Table 11), we observed increasingly negative h-index effects as coding frequency increased. Compared to non-coding authors, researchers were associated with progressively lower h-indices: occasional code contributors showed a ~27.3% lower h-index (p < 0.001), majority code contributors demonstrated a ~53.5% lower h-index (p < 0.001), and always coding authors exhibited a ~62.1% lower h-index (p < 0.001).

When controlling for author position (Table 12), we found a general pattern of reduced h-indices with increased code contribution, with one notable exception. Occasional coding first authors were associated with a ~14.9% higher h-index (p < 0.001), while always coding first authors saw a ~21.6% reduction compared to non-coding first authors (p < 0.001). For middle and last authors, the pattern was more consistently negative. Middle authors who occasionally coded showed a ~26.6% lower h-index (p < 0.001), and those always coding demonstrated a ~52.9% lower h-index (p < 0.001). Similarly, last authors who occasionally coded experienced a ~13.1% lower h-index (p < 0.001), with always coding authors showing a ~45.7% lower h-index (p < 0.001).

When controlling for research domain (Table 13), majority coding scientists showed significant h-index reductions across all domains. Health sciences researchers saw the most dramatic reduction at ~76.5% (p < 0.001), followed by physical sciences at ~52.3% ($p < 0.001$), social sciences at ~51.4% ($p < 0.001$), and life sciences at ~47.1% ($p < 0.001$).

Analyzing by common article type (Table 14) revealed similar patterns. For authors primarily publishing preprints, the h-index reductions were substantial: ~25.6% for occasional coding, ~53.5% for majority coding, and ~62.9% for always coding authors. Authors primarily publishing software articles showed slightly different but still significant reductions: ~33.1% for majority coding and ~33.0% for always coding authors.

Taken as a whole, these findings indicate that the more frequently an author contributes code, the lower their h-index is likely to be relative to their peers, with the notable exception of first authors who occasionally contribute code. What makes these results particularly striking is that each of our models includes publication count as an input feature, suggesting that these h-index reductions persist even when accounting for total research output.

5 Discussion

RE: article citations

- Our findings that code contribution and distribution among research teams has only a modest relationship with citations might reflect several underlying factors.
  - First, the relationship between code contributions and research impact may be complex and indirect - while code might enhance research reproducibility or utility, this may not directly translate into increased citations
  - Second, current citation practices may not adequately capture or credit code contributions, as authors might reference papers without explicitly acknowledging the associated code.
  - Third, the quality and significance of code contributions likely varies substantially across papers, making it difficult to detect strong aggregate effects.
  - Finally, while we have attempted to find and match as many code contributing members of research teams as possible, we must acknowledge that there may be two code-centric reasons why we have not found more code contributors:
    - First, the code may have been developed by multiple individuals but only a single individual uploaded or committed the code to a repository, thereby removing the opportunity for us to link the code to the other contributors.
    - Second, we are primarily analyzing "analysis" repositories, which may not contain the full codebase for a project. This is particularly true for projects which separate repositories for tools, libraries, and infrastructures, than the single analysis code repository.

6 References

AlShebli, Bedoor, Shahan Ali Memon, James A Evans, and Talal Rahwan. 2024. "China and the US Produce More Impactful AI Research When Collaborating Together." *Scientific Reports* 14 (1): 28576.

Biagioli, Mario, and Peter Galison. 2014. *Scientific Authorship: Credit and Intellectual Property in Science*. Routledge.

Brand, Amy, Liz Allen, Micah Altman, Marjorie Hlava, and Jo Scott. 2015. "Beyond Authorship: Attribution, Contribution, Collaboration, and Credit." *Learned Publishing* 28 (2).

Brunner, Ursin, and Kurt Stockinger. 2020. "Entity Matching with Transformer Architectures - a Step Forward in Data Integration." In *International Conference on Extending Database Technology*.

Cao, Hancheng, Jesse Dodge, Kyle Lo, Daniel A. McFarland, and Lucy Lu Wang. 2023. "The Rise of Open Science: Tracking the Evolution and Perceived Value of Data and Methods Link-Sharing Practices." *ArXiv* abs/2310.03193.

Carver, Jeffrey C., Nic Weber, Karthik Ram, Sandra Gesing, and Daniel S. Katz. 2022. "A Survey of the State of the Practice for Research Software in the United States." *PeerJ Computer Science* 8.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding." *CoRR* abs/1810.04805. http://arxiv.org/abs/1810.04805.

Edwards, Paul N, Steven J Jackson, Melissa K Chalmers, Geoffrey C Bowker, Christine L Borgman, David Ribes, Matt Burton, and Scout Calvert. 2013. "Knowledge Infrastructures: Intellectual Frameworks and Research Challenges."

Fan, Jianqing, Fang Han, and Han Liu. 2014. "Challenges of Big Data Analysis." *National Science Review* 1 (2): 293–314. https://doi.org/10.1093/nsr/nwt032.

Franceschet, Massimo, and Antonio Costantini. 2010. "The Effect of Scholar Collaboration on Impact and Quality of Academic Papers." *J. Informetrics* 4: 540–53. https://api.semanticscholar.org/CorpusID:17315541.

Gøtzsche, Peter C, Asbjørn Hróbjartsson, Helle Krogh Johansen, Mette T Haahr, Douglas G Altman, and An-Wen Chan. 2007. "Ghost Authorship in Industry-Initiated Randomised Trials." *PLoS Medicine* 4 (1): e19.

Haak, Laurel L, Martin Fenner, Laura Paglione, Ed Pentz, and Howard Ratner. 2012. “ORCID: A System to Uniquely Identify Researchers.” *Learned Publishing* 25 (4): 259–64.

Haeussler, Carolin, and Henry Sauermann. 2013. “Credit Where Credit Is Due? The Impact of Project Contributions and Social Factors on Authorship and Inventorship.” *Research Policy* 42 (3): 688–703. https://doi.org/10.1016/j.respol.2012.09.009.

Hampton, Stephanie E, Carly A Strasser, Joshua J Tewksbury, Wendy K Gram, Amber E Budden, Archer L Batcheller, Clifford S Duke, and John H Porter. 2013. “Big Data and the Future of Ecology.” *Frontiers in Ecology and the Environment* 11 (3): 156–62.

Hasselbring, Wilhelm, Stephan Druskat, Jan Bernoth, Philine Betker, Michael Felderer, Stephan Ferenz, Anna-Lena Lamprecht, Jan Linxweiler, and Bernhard Rumpe. 2024. “Toward Research Software Categories.” https://arxiv.org/abs/2404.14364.

Hata, Hideaki, Jin L. C. Guo, Raula Gaikovina Kula, and Christoph Treude. 2021. “Science-Software Linkage: The Challenges of Traceability Between Scientific Knowledge and Software Artifacts.” *ArXiv* abs/2104.05891.

He, Pengcheng, Jianfeng Gao, and Weizhu Chen. 2021. “DeBERTaV3: Improving DeBERTa Using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing.” https://arxiv.org/abs/2111.09543.

He, Pengcheng, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. “DEBERTA: DECODING-ENHANCED BERT WITH DISENTANGLED ATTENTION.” In *International Conference on Learning Representations*. https://openreview.net/forum?id=XPZIaotutsD.

Heroux, Michael A. 2022. “Research Software Science: Expanding the Impact of Research Software Engineering.” *Computing in Science & Engineering* 24 (6): 22–27. https://doi.org/10.1109/MCSE.2023.3260475.

Howison, James, Ewa Deelman, Michael J. McLennan, Rafael Ferreira da Silva, and James D. Herbsleb. 2015. “Understanding the Scientific Software Ecosystem and Its Impact: Current and Future Measures.” *Research Evaluation* 24 (4): 454–70. https://doi.org/10.1093/reseval/rvv014.

Jeong, Yuna, and Eunhui Kim. 2022. “Scideberta: Learning Deberta for Science Technology Documents and Fine-Tuning Information Extraction Tasks.” *IEEE Access* 10: 60805–13.

Jin, Xiaolong, Benjamin W Wah, Xueqi Cheng, and Yuanzhuo Wang. 2015. “Significance and Challenges of Big Data Research.” *Big Data Research* 2 (2): 59–64.

Júnior, Edilson Anselmo Corrêa, Filipi Nascimento Silva, Luciano da Fontoura Costa, and Diego Raphael Amancio. 2016. “Patterns of Authors Contribution in Scientific Manuscripts.” *J. Informetrics* 11: 498–510. https://api.semanticscholar.org/CorpusID:9967627.

Katz, Daniel S., Neil P. Chue Hong, Tim Clark, August Muench, Shelley Stall, Daina R. Bouquin, Matthew Cannon, et al. 2020. “Recognizing the Value of Software: A Software Citation Guide.” *F1000Research* 9.

Kelley, Aidan, and Daniel Garijo. 2021. “A Framework for Creating Knowledge Graphs of Scientific Software Metadata.” *Quantitative Science Studies* 2: 1423–46.

Krafczyk, Matthew, August Shi, Adhithya Bhaskar, Darko Marinov, and Victoria Stodden. 2019. “Scientific Tests and Continuous Integration Strategies to Enhance Reproducibility in the Scientific Software Context.” In *Proceedings of the 2nd International Workshop on Practical Reproducible Evaluation of Computer Systems*, 23–28. P-RECS ’19. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3322790.3330595.

Larivière, Vincent, Nadine Desrochers, Benoît Macaluso, Philippe Mongeon, Adèle Paul-Hus, and Cassidy R Sugimoto. 2016. “Contributorship and Division of Labor in Knowledge Production.” *Social Studies of Science* 46 (3): 417–35. https://doi.org/10.1177/0306312716650046.

Larivière, Vincent, Yves Gingras, Cassidy R. Sugimoto, and Andrew Tsou. 2014. “Team Size Matters: Collaboration and Scientific Impact Since 1900.” *Journal of the Association for Information Science and Technology* 66. https://api.semanticscholar.org/CorpusID:13505127.

Larivière, Vincent, David Pontille, and Cassidy R. Sugimoto. 2020. “Investigating the Division of Scientific Labor Using the Contributor Roles Taxonomy (CRediT).” *Quantitative Science Studies*, 1–18.

Li, Kai, Chenwei Zhang, and Vincent Larivière. 2023. “Are Research Contributions Assigned Differently Under the Two Contributorship Classification Systems in PLoS ONE?” *ArXiv* abs/2310.11687. https://api.semanticscholar.org/CorpusID:264289300.

Li, Yuliang, Jinfeng Li, Yoshihiko Suhara, AnHai Doan, and Wang Chiew Tan. 2020. “Deep Entity Matching with Pre-Trained Language Models.” *Proceedings of the VLDB Endowment* 14: 50–60.

Liu, Xin, Chengjing Zhang, and Jiang Li. 2023. “Conceptual and Technical Work: Who Will Disrupt Science?” *Journal of Informetrics* 17 (3): 101432. https://doi.org/https://doi.org/10.1016/j.joi.2023.101432.

Lu, Chao, Yingyi Zhang, Yong-Yeol Ahn, Ying Ding, Chenwei Zhang, and Dandan Ma. 2019. “Cocontributorship Network and Division of Labor in Individual Scientific Collaborations.” *Journal of the Association for Information Science and Technology* 71: 1162–78. https://api.semanticscholar.org/CorpusID:208267519.

Mayernik, Matthew S, David L Hart, Keith E Maull, and Nicholas M Weber. 2017. “Assessing and Tracing the Outcomes and Impact of Research Infrastructures.” *Journal of the Association for Information Science and Technology* 68 (6): 1341–59.

Meirelles, Paulo, Carlos Santos Jr, João Miranda, Fabio Kon, Antonio Terceiro, and Christina Chavez. 2010. "A Study of the Relationships Between Source Code Metrics and Attractiveness in Free Software Projects." In *2010 Brazilian Symposium on Software Engineering*, 11–20. IEEE.

Merow, Cory, Brad L. Boyle, Brian J. Enquist, Xiao Feng, Jamie M. Kass, Brian Salvin Maitner, Brian McGill, et al. 2023. "Better Incentives Are Needed to Reward Academic Software Development." *Nature Ecology & Evolution* 7: 626–27.

Milewicz, Reed, Gustavo Pinto, and Paige Rodeghero. 2019. "Characterizing the Roles of Contributors in Open-Source Scientific Software Projects." In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*, 421–32. https://doi.org/10.1109/MSR.2019.00069.

Muna, Demitri, Michael Alexander, Alice Allen, Richard Ashley, Daniel Asmus, Ruyman Azzollini, Michele Bannister, et al. 2016. "The Astropy Problem." https://arxiv.org/abs/1610.03159.

Ni, Chaoqun, Elise Smith, Haimiao Yuan, Vincent Larivière, and Cassidy R. Sugimoto. 2021. "The Gendered Nature of Authorship." *Science Advances* 7 (36): eabe4639. https://doi.org/10.1126/sciadv.abe4639.

Philippe, Olivier, Martin Hammitzsch, Stephan Janosch, Anelda van der Walt, Ben van Werkhoven, Simon Hettrick, Daniel S. Katz, et al. 2019. "Softwaresaved/International-Survey: Public Release for 2018 Results." Zenodo. https://doi.org/10.5281/zenodo.2585783.

Ram, Karthik. 2013. "Git Can Facilitate Greater Reproducibility and Increased Transparency in Science." *Source Code for Biology and Medicine* 8: 1–8.

Sanh, Victor, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. "DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter." *ArXiv* abs/1910.01108.

Sauermann, Henry, and Carolin Haeussler. 2017. "Authorship and Contribution Disclosures." *Science Advances* 3 (11): e1700404. https://doi.org/10.1126/sciadv.1700404.

Sharma, Nitesh Kumar, Ram Ayyala, Dhrithi Deshpande, Yesha Patel, Viorel Munteanu, Dumitru Ciorba, Viorel Bostan, et al. 2024. "Analytical Code Sharing Practices in Biomedical Research." *PeerJ Computer Science* 10: e2066.

Stankovski, Aleksandar, and Daniel Garijo. 2024. "RepoFromPaper: An Approach to Extract Software Code Implementations from Scientific Publications." In *NSLP*.

Stodden, Victoria, Peixuan Guo, and Zhaokun Ma. 2013. "Toward Reproducible Computational Research: An Empirical Analysis of Data and Code Policy Adoption by Journals." *PloS One* 8 (6): e67111.

Tran, Hanh Thi Hong, Nishan Chatterjee, Senja Pollak, and Antoine Doucet. 2024. "DeBERTa Beats Behemoths: A Comparative Analysis of Fine-Tuning, Prompting, and PEFT Approaches on LegalLensNER." In *Proceedings of the Natural Legal Language Processing Workshop 2024*, 371–80.

Trisovic, Ana, Matthew K. Lau, Thomas Pasquier, and Mercè Crosas. 2021. "A Large-Scale Study on Research Code Quality and Execution." *Scientific Data* 9.

Trujillo, Milo Z, Laurent Hébert-Dufresne, and James Bagrow. 2022. "The Penumbra of Open Source: Projects Outside of Centralized Platforms Are Longer Maintained, More Academic and More Collaborative." *EPJ Data Science* 11 (1): 31.

Westner, Britta U., Daniel R. McCloy, Eric Larson, Alexandre Gramfort, Daniel S. Katz, Arfon M. Smith, and invited co-signees. 2024. "Cycling on the Freeway: The Perilous State of Open Source Neuroscience Software." *ArXiv* abs/2403.19394.

Wyss, Elizabeth, Lorenzo De Carli, and Drew Davidson. 2023. "(Nothing but) Many Eyes Make All Bugs Shallow." In *Proceedings of the 2023 Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses*, 53–63. SCORED '23. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3605770.3625216.

Yu, Liqiang, Bo Liu, Qunwei Lin, Xinyu Zhao, and Chang Che. 2024. "Semantic Similarity Matching for Patent Documents Using Ensemble BERT-Related Model and Novel Text Processing Method." *arXiv Preprint arXiv:2401.06782*.

7 Appendix

7.1 Extended Modeling and Analysis Results and Supporting Tables

7.1.1 Full Comparison of Models and Optional Features for Author-Developer-Account Matching

Table 4: Comparison of Models for Author-Developer-Account Matching

Optional Feats.	Model	Accuracy	Precision	Recall	F1
name	deberta	0.984	0.938	0.950	0.944
name, email	bert-multilingual	0.984	0.938	0.950	0.944
name, email	deberta	0.982	0.907	0.975	0.940
name	bert-multilingual	0.982	0.938	0.938	0.938
name	distilbert	0.978	0.936	0.912	0.924
name, email	distilbert	0.978	0.936	0.912	0.924
email	deberta	0.957	0.859	0.838	0.848
email	bert-multilingual	0.950	0.894	0.738	0.808
n/a	deberta	0.946	0.847	0.762	0.803
n/a	bert-multilingual	0.941	0.862	0.700	0.772
n/a	distilbert	0.856	0.000	0.000	0.000
email	distilbert	0.856	0.000	0.000	0.000

7.1.2 Linear Models for Software Development Dynamics Within Research Teams

Table 5: Article Citations by Code Contributorship of Research Team

Variable	coef	P>|z|	[0.025	0.975]
const ***	**0.99**	**0.00**	**0.94**	**1.04**
Total Authors ***	**0.07**	**0.00**	**0.06**	**0.08**
Code-Contrib. Authors ***	**0.04**	**0.00**	**0.02**	**0.06**
Code-Contrib. Non-Authors	-0.00	0.80	-0.01	0.01
Years Since Publication ***	**0.39**	**0.00**	**0.38**	**0.40**

Table 6: Article Citations by Code Contributorship of Research Team Controlled by Open Access Status

Variable	coef	P>|z|	[0.025	0.975]
const ***	**0.61**	**0.00**	**0.49**	**0.73**
Total Authors ***	**0.07**	**0.00**	**0.06**	**0.08**
Code-Contrib. Authors	0.05	0.24	-0.04	0.14
Code-Contrib. Non-Authors	0.00	0.96	-0.03	0.03
Years Since Publication ***	**0.38**	**0.00**	**0.37**	**0.39**
Is Open Access ***	**0.42**	**0.00**	**0.30**	**0.54**
Code-Contrib. Authors * Is Open Access	-0.01	0.82	-0.10	0.08
Code-Contrib. Non-Authors * Is Open Access	-0.00	0.94	-0.03	0.03

Table 7: Article Citations by Code Contributorship of Research Team Controlled by Domain

Variable	coef	P>|z|	[0.025	0.975]
const ***	**0.88**	**0.00**	**0.75**	**1.01**
Total Authors ***	**0.07**	**0.00**	**0.06**	**0.08**
Code-Contrib. Authors	0.03	0.60	-0.08	0.13
Code-Contrib. Non-Authors	0.01	0.61	-0.04	0.07
Years Since Publication ***	**0.40**	**0.00**	**0.39**	**0.41**
Domain Life Sciences ***	**-0.21**	**0.01**	**-0.36**	**-0.06**
Domain Physical Sciences ***	**0.14**	**0.03**	**0.01**	**0.26**
Domain Social Sciences ***	**-0.18**	**0.03**	**-0.34**	**-0.02**
Code-Contrib. Authors * Domain Life Sciences	0.07	0.29	-0.06	0.19
Code-Contrib. Authors * Domain Physical Sciences	0.00	0.93	-0.10	0.11
Code-Contrib. Authors * Domain Social Sciences	0.10	0.14	-0.03	0.23
Code-Contrib. Non-Authors * Domain Life Sciences	-0.04	0.23	-0.11	0.03
Code-Contrib. Non-Authors * Domain Physical Sciences	-0.02	0.57	-0.07	0.04
Code-Contrib. Non-Authors * Domain Social Sciences	-0.04	0.31	-0.11	0.03

Table 8: Article Citations by Code Contributorship of Research Team Controlled by Article Type

Variable	coef	P>|z|	[0.025	0.975]
const ***	**0.53**	**0.00**	**0.45**	**0.61**
Total Authors ***	**0.07**	**0.00**	**0.06**	**0.08**
Code-Contrib. Authors	-0.03	0.20	-0.09	0.02
Code-Contrib. Non-Authors ***	**-0.03**	**0.00**	**-0.05**	**-0.01**
Years Since Publication ***	**0.40**	**0.00**	**0.39**	**0.41**
Article Type Research Article ***	**0.47**	**0.00**	**0.40**	**0.55**
Article Type Software Article ***	**-0.47**	**0.00**	**-0.73**	**-0.22**
Code-Contrib. Authors * Article Type Research Article ***	**0.10**	**0.00**	**0.05**	**0.16**
Code-Contrib. Authors * Article Type Software Article	-0.06	0.37	-0.19	0.07
Code-Contrib. Non-Authors * Article Type Research Article ***	**0.04**	**0.00**	**0.02**	**0.06**
Code-Contrib. Non-Authors * Article Type Software Article	0.09	0.24	-0.06	0.24

7.1.3 Post-Hoc Tests for Coding vs Non-Coding Authors by Position

Table 9: Counts of Code-Contributing Authors ('Coding') as well as Total Authors by Position and Bonferroni Corrected p-values from Post-Hoc Binomial Tests

Control	Subset	Position	Coding	Total	p
Domain	**Health Sciences**	First	1575	2401	0.000***
		Middle	540	11491	0.000***
		Last	234	2349	0.000***
	Life Sciences	First	2586	3895	0.000***
		Middle	852	11875	0.000***
		Last	491	3784	0.000***
	Physical Sciences	First	28919	41987	0.000***
		Middle	10507	111738	0.000***
		Last	3433	40410	0.000***
	Social Sciences	First	2813	4038	0.000***
		Middle	1021	9249	0.000***
		Last	411	3855	0.000***
Article Type	**Preprint**	First	13421	19523	0.000***
		Middle	5134	52925	0.000***
		Last	1493	18598	0.000***
	Research Article	First	21940	32081	0.000***
		Middle	7345	89991	0.000***
		Last	2909	31188	0.000***
	Software Article	First	532	717	0.000***
		Middle	441	1437	0.000***
		Last	167	612	0.000***
Open Access Status	**Closed Access**	First	2637	3742	0.000***
		Middle	918	10965	0.000***
		Last	279	3642	0.000***
	Open Access	First	33256	48579	0.000***
		Middle	12002	133388	0.000***
		Last	4290	46756	0.000***
Overall	**Overall**	First	35893	52321	0.000***
		Middle	12920	144353	0.000***
		Last	4569	50398	0.000***

7.1.4 Post-Hoc Tests for Coding vs Non-Coding Authors by Corresponding Status

Table 10: Counts of Code-Contributing Authors ('Coding') as well as Total Authors by Corresponding Status and Bonferroni Corrected p-values from Post-Hoc Binomial Tests

Control	Subset	Is Corresponding	Coding	Total	p
Domain	**Life Sciences**	Corresponding	1772	8019	0.000***
		Not Corresponding	2157	11535	0.000***
	Physical Sciences	Corresponding	5248	12487	0.000***
		Not Corresponding	37611	181648	0.000***
	Social Sciences	Corresponding	803	2458	0.000***
		Not Corresponding	3442	14684	0.000***
Article Type	**Preprint**	Corresponding	772	1036	0.000***
		Not Corresponding	19276	90010	0.000***
	Research Article	Corresponding	7716	27339	0.000***
		Not Corresponding	24478	125921	0.000***
	Software Article	Corresponding	213	438	1.000
		Not Corresponding	927	2328	0.000***
Open Access Status	**Closed Access**	Corresponding	253	468	0.174
		Not Corresponding	3581	17881	0.000***
	Open Access	Corresponding	8448	28345	0.000***
		Not Corresponding	41100	200378	0.000***
Overall	**Overall**	Corresponding	8701	28813	0.000***
		Not Corresponding	44681	218259	0.000***

7.1.5 Linear Models for Characterizing Code-Contributing Author H-Index

Table 11: Code-Contributing Authors H-Index by Coding Status

Variable	coef	P>|z|	[0.025	0.975]
const ***	**3.19**	**0.00**	**3.18**	**3.20**
Works Count ***	**0.00**	**0.00**	**0.00**	**0.00**
Any Coding ***	**-0.32**	**0.00**	**-0.33**	**-0.30**
Majority Coding ***	**-0.77**	**0.00**	**-0.79**	**-0.74**
Always Coding ***	**-0.97**	**0.00**	**-1.02**	**-0.92**

Table 12: Researcher H-Index by Coding Status Controlled by Most Freq. Author Position

Variable	coef	P>|z|	[0.025	0.975]
const ***	**2.38**	**0.00**	**2.31**	**2.44**
Works Count ***	**0.00**	**0.00**	**0.00**	**0.00**
Any Coding ***	**0.14**	**0.00**	**0.05**	**0.22**
Majority Coding	-0.07	0.08	-0.14	0.01
Always Coding ***	**-0.24**	**0.00**	**-0.33**	**-0.16**
Common Author Position Last ***	**1.05**	**0.00**	**0.98**	**1.11**
Common Author Position Middle ***	**0.74**	**0.00**	**0.68**	**0.81**
Any Coding * Common Author Position Last ***	**-0.28**	**0.00**	**-0.37**	**-0.19**
Any Coding * Common Author Position Middle ***	**-0.45**	**0.00**	**-0.53**	**-0.36**
Majority Coding * Common Author Position Last ***	**-0.35**	**0.00**	**-0.45**	**-0.26**
Majority Coding * Common Author Position Middle ***	**-0.61**	**0.00**	**-0.69**	**-0.52**
Always Coding * Common Author Position Last ***	**-0.37**	**0.00**	**-0.51**	**-0.22**
Always Coding * Common Author Position Middle ***	**-0.51**	**0.00**	**-0.64**	**-0.38**

Table 13: Researcher H-Index by Coding Status Controlled by Most Freq. Domain

Variable	coef	P>|z|	[0.025	0.975]
const ***	**3.32**	**0.00**	**3.29**	**3.36**
Works Count ***	**0.00**	**0.00**	**0.00**	**0.00**
Any Coding ***	**-0.39**	**0.00**	**-0.48**	**-0.31**
Majority Coding ***	**-1.45**	**0.00**	**-1.65**	**-1.25**
Always Coding ***	**-1.22**	**0.00**	**-1.58**	**-0.86**
Common Domain Life Sciences ***	**0.10**	**0.00**	**0.05**	**0.15**
Common Domain Physical Sciences ***	**-0.15**	**0.00**	**-0.18**	**-0.11**
Common Domain Social Sciences ***	**-0.16**	**0.00**	**-0.22**	**-0.10**
Any Coding * Common Domain Life Sciences	0.11	0.07	-0.01	0.22
Any Coding * Common Domain Physical Sciences	0.08	0.09	-0.01	0.17
Any Coding * Common Domain Social Sciences	0.01	0.94	-0.14	0.15
Majority Coding * Common Domain Life Sciences ***	**0.81**	**0.00**	**0.58**	**1.04**
Majority Coding * Common Domain Physical Sciences ***	**0.70**	**0.00**	**0.50**	**0.90**
Majority Coding * Common Domain Social Sciences ***	**0.72**	**0.00**	**0.46**	**0.98**
Always Coding * Common Domain Life Sciences	0.33	0.12	-0.08	0.74
Always Coding * Common Domain Physical Sciences	0.25	0.18	-0.12	0.62
Always Coding * Common Domain Social Sciences	0.30	0.17	-0.13	0.73

Table 14: Researcher H-Index by Coding Status Controlled by Most Freq. Article Type

Variable	coef	P>|z|	[0.025	0.975]
const ***	**3.10**	**0.00**	**3.09**	**3.11**
Works Count ***	**0.00**	**0.00**	**0.00**	**0.00**
Any Coding ***	**-0.30**	**0.00**	**-0.32**	**-0.27**
Majority Coding ***	**-0.77**	**0.00**	**-0.81**	**-0.73**
Always Coding ***	**-0.99**	**0.00**	**-1.07**	**-0.92**
Common Article Type Research Article ***	**0.18**	**0.00**	**0.17**	**0.20**
Common Article Type Software Article ***	**0.22**	**0.00**	**0.12**	**0.33**
Any Coding * Common Article Type Research Article	-0.02	0.15	-0.05	0.01
Any Coding * Common Article Type Software Article	0.19	0.06	-0.01	0.39
Majority Coding * Common Article Type Research Article	0.01	0.80	-0.05	0.06
Majority Coding * Common Article Type Software Article ***	**0.36**	**0.00**	**0.19**	**0.54**
Always Coding * Common Article Type Research Article	0.00	0.93	-0.10	0.10
Always Coding * Common Article Type Software Article ***	**0.37**	**0.00**	**0.15**	**0.58**

7.2 Analysis of Project Duration and Percentage Code-Contributors Who Are Authors

In our pre-registered analysis plan (https://osf.io/fc74m), we originally hypothesized about the relationship between project duration and authorship recognition. Specifically, we posited that sustained technical engagement and academic recognition might be meaningfully related, with longer project durations potentially leading to higher rates of code-contributor authorship. We saw repository histories as providing a unique opportunity to examine this relationship, leading us to hypothesize that projects with longer commit durations would be associated with higher percentages of developers receiving authorship recognition (pre-registered as H2).

However, our analysis found no evidence to support this hypothesis. When examining the relationship between a repository's commit duration and the percentage of developers who receive authorship recognition, we found no significant correlation (r = -0.00, p = n.s.). This suggests that the length of time a project has been in development has no meaningful relationship with the proportion of developers who are recognized as authors.

We ultimately decided to move this analysis to the appendix for two key methodological reasons. First, our approach of using repository-level commit duration as a proxy for individual contribution patterns proved too coarse-grained. A more precise analysis would need to examine individual-level contribution durations and patterns rather than overall project length. Second, our method did not account for the varying levels of contribution that different developers make to a repository. Simply correlating overall project duration with authorship rates fails to capture the nuanced ways that sustained, meaningful technical contributions might influence authorship decisions.

These limitations suggest potential directions for future work that could more rigorously examine the relationship between long-term technical engagement and academic recognition. Such work might benefit from more granular analysis of individual contribution patterns, perhaps incorporating measures of contribution significance and sustainability rather than just temporal duration.