Reproducibility Debt in Scientific Software

Zara Hassan
Australian National University
Canberra, Australia
zara.hassan@anu.edu.au

Christoph Treude
Singapore Management University
Singapore, Singapore
ctreude@smu.edu.sg

Graham Williams
Australian National University
Canberra, Australia
graham.williams@anu.edu.au

Michael Norrish
Australian National University
Canberra, Australia
michael.norrish@anu.edu.au

Alex Potanin
Australian National University
Canberra, Australia
alex.potanin@anu.edu.au

Abstract

Reproducibility Debt (RpD) refers to accumulated technical and organisational issues in scientific software that hinder the ability to reproduce research results. While reproducibility is essential to scientific integrity, RpD remains poorly defined and under-addressed. This study introduces a formal definition of RpD and investigates its causes, effects, and mitigation strategies using a mixed-methods approach involving a systematic literature review (214 papers), interviews (23 practitioners), and a global survey (59 participants). We identify seven categories of contributing issues, 75 causes, 110 effects, and 61 mitigation strategies. Findings are synthesised into a cause-effect model and supported by taxonomies of team roles and software types. This work provides conceptual clarity and practical tools to help researchers, developers, and institutions understand and manage RpD, ultimately supporting more sustainable and reproducible scientific software.

CCS Concepts: • **Software and its engineering → Open source model**; **Reusability**; **Traceability**.

Keywords: Reproducibility, Technical Debt, Scientific Software, Scientific Computing, Computational Reproducibility

ACM Reference Format:
Zara Hassan, Christoph Treude, Graham Williams, Michael Norrish, and Alex Potanin. 2025. Reproducibility Debt in Scientific Software. In *Companion Proceedings of the 2025 ACM SIGPLAN International Conference on Systems, Programming, Languages, and Applications: Software for Humanity (SPLASH Companion '25), October 12–18, 2025, Singapore, Singapore.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3758316.3765482

SPLASH Companion '25, Singapore, Singapore

ACM ISBN 979-8-4007-2141-0/25/10
https://doi.org/10.1145/3758316.3765482

1 Introduction

Technical Debt (TD) refers to short-term software decisions that incur long-term costs, reducing maintainability and quality [1]. While widely studied in traditional software engineering, TD is also prevalent in scientific software, i.e, tools developed by domain experts, often without formal software engineering training [5]. This is particularly problematic as scientific software today underpins critical research across multiple disciplines.

A key challenge in scientific computing is reproducibility; the ability to replicate and build upon prior results [3]. Yet, over 70% of researchers report being unable to reproduce published findings, contributing to what is now known as the 'reproducibility crisis' [2]. In software-intensive research, failures in reproducibility often stem from technical issues, such as missing dependencies, undocumented workflows, or non-deterministic behaviour. We conceptualize these technical barriers as Reproducibility Debt (RpD) [3], a type of TD specific to scientific software that compromises reproducibility. While the term has been briefly mentioned in contexts like machine learning and data-intensive research, there is currently no consolidated definition, classification, or mitigation framework.

To address this gap, we conducted a mixed-methods study consisting of a systematic literature review (SLR) of 214 papers [4], interviews with 23 scientific software practitioners, and a global survey *(InsightRpD)* with 59 participants. We identified 75 causes and 110 effects of RpD, and developed a probabilistic cause-effect model to represent these relationships. Our findings also consolidate mitigation strategies and provide a shared vocabulary for understanding and managing RpD. This work contributes both conceptual clarity and practical tools for improving the sustainability and reliability of scientific software.

2 Methodology

The study explored RpD in scientific software using a mixed-methods approach, guided by three research questions (RQs): ***RQ1:*** *What is RpD in scientific software and what are the main categories of issues that contribute to it?*; ***RQ2:*** *What are the underlying causes and effects of RpD in scientific software?*;

RQ3: *What are the best practices for the management and mitigation of RpD in scientific software projects?*

To answer RQ1, we conducted our SLR [4] following established guidelines. The search was performed across six major digital libraries and Google Scholar using a carefully constructed search string. Papers were selected through a three-stage filtering process (metadata, abstract, and full-text evaluation), and assessed using quality criteria. The final pool included 206 primary studies, with an additional 8 papers added from a 2024 update. This review allowed us to formally define RpD and construct a high-level taxonomy of issues contributing to it, providing the foundation for the rest of the study. The SLR also partially contributed to answering RQ2 and RQ3, by highlighting 37 causes, 63 effects, and mitigation strategies discussed in the literature.

To address RQ2 and RQ3 in greater depth, we conducted semi-structured interviews with scientific software practitioners from a large government research organisation. We designed a 22-question interview guide informed by the SLR and the InsighTD framework [6]. Participants represented five key roles involved in scientific software development across various stages, from data acquisition to analysis and reporting. These interviews provided rich, practice-based insights into how and why RpD occurs, and how teams manage or prevent it. Finally, to validate and extend our findings at scale, we developed the *InsightRpD* survey tool, using a design and validation process based on the InsighTD methodology [6]. The survey integrated insights from both the SLR and interviews and was refined through expert reviews and a pilot study. It was then disseminated via social media platforms such as LinkedIn and X (formerly Twitter). The survey responses helped to triangulate findings for RQ2 and RQ3, providing broader perspectives on causes, effects, and mitigation strategies from a wider community of scientific software practitioners.

3 Results

By answering RQ1, we are able to present the first formal definition of RpD based on data in existing literature: *"Reproducibility Debt (RpD) is a type of technical debt primarily impacting scientific software. It refers to the accumulative issues and challenges that hinder the ability to reproduce scientific outcomes, stemming from challenges inherent in scientific software code and data, including development, organisation, dissemination, and documentation."*. From the analysis of 214 primary studies, we categorised the contributing issues into seven main categories: human-centric, data-centric, documentation-centric, tools and infrastructure, code-centric, legal, and versioning issues. Human and organisational factors were the most frequently cited, followed by issues related to data and documentation. This taxonomy offers a structured vocabulary to describe and compare reproducibility challenges across scientific software projects.

By combining findings from the SLR, interviews, and survey, we identified 75 distinct causes and 110 effects of RpD. These include socio-technical challenges such as *lack of formal training, insufficient documentation*, and *time or funding constraints*. Effects include *reduced software reuse, inability to validate results,* and *degraded research quality*. We synthesised these findings into probabilistic cause-effect diagrams to visualise the relationships between frequently cited causes and their downstream consequences, offering practitioners a tool for understanding and mitigating RpD. Our analysis identified 61 mitigation strategies (RQ3), most of which stem from real-world practitioner experience. To support more context-aware strategies, we introduce two supporting taxonomies: one outlining contributor roles in scientific software teams, and another classifying real-world scientific software types, both aimed at enabling more targeted tool and policy design.

4 Relevance and Potential Impact

This study provides an evidence-based understanding of Reproducibility Debt (RpD) in scientific software by identifying its causes, effects, and mitigation strategies. The developed taxonomies and cause-effect diagrams support targeted interventions and informed decision-making, emphasising the need for organisational support and cultural change. Future work should validate these findings across broader contexts, develop practical tools for monitoring and managing RpD, and examine structural and policy influences.

References

[1] Nicolli S.R. Alves, Thiago S. Mendes, Manoel G. de Mendonça, Rodrigo O. Spínola, Forrest Shull, and Carolyn Seaman. 2016. Identification and Management of Technical Debt: A Systematic Mapping Study. *Information and Software Technology* 70 (2016), 100–121. doi:10.1016/j.infsof.2015.10.008

[2] Monya Baker. 2016. 1,500 Scientists Lift the Lid on Reproducibility. *Nature* 533, 7604 (2016), 452–454. doi:10.1038/533452a

[3] Zara Hassan, Christoph Treude, Michael Norrish, Graham Williams, and Alex Potanin. 2024. Reproducibility Debt: Challenges and Future Pathways. In *Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering* (Porto de Galinhas, Brazil) *(FSE 2024)*. Association for Computing Machinery, New York, NY, USA, 462–466. doi:10.1145/3663529.3663778

[4] Zara Hassan, Christoph Treude, Michael Norrish, Graham Williams, and Alex Potanin. 2025. Characterising reproducibility debt in scientific software: A systematic literature review. *Journal of Systems and Software* 222 (2025), 112327. doi:10.1016/j.jss.2024.112327

[5] Arne N. Johanson and Wilhelm Hasselbring. 2018. Software Engineering for Computational Science: Past, Present, Future. *Computing in Science & Engineering* 20 (2018), 90–109. doi:10.1109/MCSE.2018.021651343

[6] Nicolli Rios, Rodrigo Spínola, Manoel Mendonça, and Carolyn Seaman. 2020. The practitioners' point of view on the concept of technical debt and its causes and consequences: a design for a global family of industrial surveys and its first results from Brazil. *Empirical Software Engineering* 25 (09 2020). doi:10.1007/s10664-020-09832-9

Received 2025-08-17; accepted 2025-08-24