Advancing Sustainable Communities in Scientific OSS: A Replication Study with Astropy

Jiayi Sun*, Aarya Patil†, Youhai Li‡, Jin L.C. Guo§, Shurui Zhou*
*University of Toronto, †Max Planck Institute for Astronomy, ‡Carnegie Mellon University, §McGill University

***Abstract*—Open-source software (OSS) fosters transparency and collaboration across various domains. Scientific OSS, built on the open-source model, has been especially valuable in supporting scientific discovery and the corresponding research communities. However, ensuring the sustainability of the community by attracting newcomers and retaining existing contributors is recognized as a major challenge for the long-term success of OSS. While previous research has studied sustainability in OSS broadly, the active involvement of scientists—many of whom lack formal training in professional software engineering—and the presence of highly interdependent projects that create a multi-project ecosystem with interconnected communities may introduce unique challenges that differ from those in commonly studied OSS contexts. In this study, we explore sustainability challenges and opportunities within the scientific OSS community, aiming to understand how these aspects align with or differ from those in previously researched OSS communities. Through a survey-based replication study in the *Astropy Project*—a widely-used OSS ecosystem in astronomy—we gathered insights from disengaged contributors regarding their motivations, reasons for disengagement, and suggestions for improving community sustainability. Our findings reveal key motivations driving contributions to scientific OSS, identify barriers to sustained engagement, and propose strategies to address these challenges, highlighting areas for future research.**

***Index Terms*—scientific software, open-source software, sustainable community**

I. Introduction

Scientific discoveries in fields like biology, physics, and astronomy increasingly rely on software to support data analysis and computational tasks. Scientific software (also known as research software), encompassing source code, algorithms, scripts, workflows, and executables created for scientific purposes [1], plays a vital role in advancing these efforts. Building on the success of open-source models, many scientific communities have adopted open-source development for their projects, creating ecosystems like ImageJ for biomedical imaging [2] and Biopython for bioinformatics [3]. These efforts have boosted research reproducibility and reuse, enabling scientific breakthroughs like gravitational wave research [4] and COVID-19 molecular simulations [5]. As reliance on software grows, ensuring its sustainability—so scientists can understand, replicate, and expand on results—has become a major priority [6], [7], with increasing global investments reflecting this need [8]–[15].

A widely recognized and persistent challenge within the OSS communities is enhancing sustainability, particularly in motivating new contributors to join and encouraging existing members to remain actively engaged over time. Although prior research has investigated factors contributing to contributor disengagement and suggested approaches to improve the sustainability of the OSS communities, these efforts have primarily centered on non-scientific OSS. Scientific OSS communities, however, differ from general OSS communities in several respects, notably in (1) the diversity of team members' backgrounds and (2) the complexity of interdependent relationships with other projects in the larger ecosystem. Specifically, due to the interdisciplinary nature of scientific software development, one primary challenge highlighted in previous research is that scientists generally lack sufficient training in software engineering (SE), which compromises the quality, usability, and sustainability of scientific software [16]–[18]. This challenge becomes increasingly important as the software evolves to be more complex. To address such a challenge, experienced software engineers are frequently brought on to the team to handle crucial engineering tasks [19]–[22]. However, *interdisciplinary collaboration* between scientists and software engineers can be difficult because the two groups often have different objectives (*research* vs. *development*), educational backgrounds, and mindsets [20], [23]–[25], thereby hindering the software development process [26]–[29]. Furthermore, due to the interdependencies among projects within the broader ecosystem of similar scientific domains, OSS communities are interconnected and often overlap rather than function in isolation—a dynamic that remains underexplored from a sustainability perspective. Consequently, these differences prompt an inquiry into (1) whether scientific OSS encounters comparable challenges related to community sustainability; and (2) whether the existing solutions in OSS (e.g., effective governance of OSS community, sponsors and donations to support developers [30], [31]) can be directly applied to scientific OSS communities.

While the sustainability of OSS community can manifest in different aspects, we focus on the community retention aspect in current study as understanding why contributors stay—or disengage—can provide actionable insights for improving community structures, governance, and support mechanisms. Therefore, in this study, to understand the challenges and opportunities regarding sustaining scientific OSS communities, we explore the following research questions:

- **RQ1 (Motivation)**: *What are the motivations for contributing to scientific OSS?*
- **RQ2 (Disengagement)**: *What are the main factors for disengaging from scientific OSS?*
- **RQ3 (Suggestions for improvement)**: *How to improve*

the sustainability of scientific OSS communities from practitioners' perspectives?

To address these RQs, we conducted a replication study [32], [33] using a survey with disengaged contributors of the *Astropy* project to understand their motivations for contributing and reasons for disengagement. We selected the *Astropy* project—an ecosystem of open-source Python libraries in astronomy with over 10 years of development history—due to its established community structure and diversity of contributors across 51 interoperable libraries, each maintained as separate GitHub repositories (as of August 2022) [34]. This diversity and scale provide an ideal environment for examining community sustainability challenges and patterns in contributor engagement, potentially offering insights applicable to other scientific OSS communities.

Our findings reveal that while scientific OSS communities in the *Astropy* ecosystem share common challenges with general OSS, such as onboarding new contributors and maintaining engagement, they also face unique sustainability issues. Unlike general OSS, where motivations have recently shifted toward social factors like *altruism*, scientific OSS contributors are still predominantly driven by *own-use*, reflecting the research-driven nature of these projects. This focus on *own-use* also explains why contributors often disengage as their research interests change or due to the steep learning curve, suggesting that community-centered retention strategies alone may be insufficient.

Drawing from our findings, we propose future research and tool development focused on supporting the long-term viability of the *Astropy* project and scientific OSS communities more broadly, such as developing quantitative metrics for academic acknowledgment within OSS and improving SE infrastructure to facilitate contribution workflows.

II. Related Work

A. Scientific Software Development & Scientific OSS

Research has identified key technical obstacles in scientific software development, from design [35] to testing [36] and release [37]. Scientific software often struggles to meet quality standards, particularly in traceability [38], [39], reproducibility [16], [40], [41], and sustainability [6], [7]. Many funding bodies now emphasize the need for long-term sustainable software [42]–[46], recommending improvements include training scientists in programming [16], [18] and strengthening maintenance infrastructure [26]. However, reproducibility and sustainability challenges persist [41], [47] despite ongoing efforts.

In recent years, scientific OSS communities also publish papers to document their experiences, share insights on project and community management, and promote citation to enhance contributor recognition. For example, the rOpensci community [48] shared strategies for building sustainable communities through social media engagement, workshops, and reproducibility-focused workflows [49]. The Astropy community published three major papers [50]–[52] over nine years, covering software features, community efforts, governance, and funding. Similarly, the 2019 CS&S report [53] highlighted sustainability themes like funding and leadership from surveys of research-driven OSS communities.

Moreover, within the context of scientific OSS, software engineering (SE) research communities have begun to explore scientific OSS from various angles, including collaboration practices among various roles based on contributors' academic seniority [54], and technical debt present in the documentation of scientific OSS [55].

Complementing the body of research published across various communities, our study investigates deeper into the sustainability challenges faced by scientific OSS communities, specifically through an examination of the experiences of disengaged contributors to identify challenges that existing community sustainability strategies have yet to address. We explore community sustainability from both contributor and ecosystem perspectives, offering a more comprehensive understanding of these challenges.

B. Sustaining OSS Communities in General

1) Motivation for contributing to OSS.

Because of the voluntary nature of contributions, OSS projects have been facing sustainability challenges regarding retaining existing contributors and attracting newcomers [56]. Prior studies investigated the motivation of different types of contributors, including long-term core contributors and peripheral contributors who make casual contributions [57]. Various types of motivations have been observed, including (1) intrinsic motivations, such as *altruism*, *kinship*, *fun*, and *ideology*; (2) internalized extrinsic motivations such as *reputations*, *reciprocity*, *learning*, and *own-use* (e.g., "scratch one's own itch") [58]–[60]; and (3) extrinsic motivations, such as career and pay [61]. In 2021, Gerosa et al. revisited the motivations for contributions previously identified in research on OSS projects to examine how motivation has evolved in light of the emergence of platforms like GitHub [62]. They found a growing emphasis on social motivations (e.g., *altruism* to help others and community, or *reputation* to seek recognition for contribution) over the years, shifting from the *own-use* to address personal needs.

2) Reasons for disengaging the OSS community

Still, when contributing to OSS projects, the disengagement of existing contributors due to reasons like heavy workloads [63] or hostile community culture [64] poses another challenge to the sustainability of OSS projects. To investigate why established OSS contributors disengage, Miller et al. [63] surveyed contributors who had ceased activity on GitHub, finding that the most common reasons of disengagement were occupational, such as job changes or increased workload, but social and technical reasons were also cited, including loss of interest or platform issues.

3) Suggestions for sustaining OSS community.

Moreover, newcomers face both social barriers (e.g., lack of social interaction with maintainers) and technical barriers (e.g., lack of necessary previous knowledge, difficulty finding tasks to work on) [65] when it comes to contributing to OSS projects. To lower the entry barrier for contribution, best

practices such as good first issues [66], mentoring [67], and the Google Summer of Code programs [68] have been adopted by OSS projects to help onboard newcomers. At the same time, many projects have leveraged donations and funding from organizations to better support the maintenance process and retain existing contributors [31], [69]. Other practices such as the Code of Conduct are also widely adopted to facilitate welcoming and healthy community cultures [70].

To assess whether this observation holds within scientific OSS communities, our replication study compares the motivations identified by Gerosa et al. [62] with the reasons for disengagement observed by Miller et al. [63]. We highlight the unique characteristics of scientific OSS communities and explore strategies for improving community sustainability.

C. *Differences between Scientific OSS and OSS in general*

As a subset of the OSS communities in general, scientific OSS face the commonly mentioned sustainability challenges, such as attracting newcomers [71], [72], retaining contributors [73], scaling up the communities [74]. Nevertheless, it is unknown whether the concerns in sustaining communities are common to both scientific OSS and general OSS. Additionally, it is unclear whether the solutions proposed in previous work are still applicable, considering the distinct characteristics between scientific OSS and general OSS, such as:

- Various motivations drive contributions. Academic credit is assigned to scientific contributions, while open-source contributions are gauged by the effort put into software-related activities [75].
- It is more challenging to onboard and retain contributors to scientific projects than general OSS as explored in academic research and grey literature studies [51], [76].
- Different funding models, such as research institute affiliations and grant funding, differ significantly from the predominantly voluntary approach commonly observed in general OSS projects [24], [26]
- Distinct stakeholder compositions, exemplified by a small community affiliated with a specific research domain versus a diverse contributor background, contribute to the differentiation between the two [24], [26]

Our study can be considered as a *conceptual replication* [32], [33] of prior work, which concentrates on the sustainability challenges of general OSS communities [62], focusing on the motivation of contribution and the reasons for disengagement [63]. Specifically, our RQ1 replicates Gerosa et al.'s work [62] on OSS contribution motivation, but with a focus on scientific OSS communities. While Gerosa et al. surveyed contributors from diverse OSS backgrounds using both Likert-scale and open-ended questions to analyze motivation shifts, our study targets scientific OSS contributors and relies on open-ended questions for richer qualitative insights. For RQ2, we replicate Miller et al.'s study [63] on reasons for disengagement from OSS, adapting it to the scientific OSS context. While Miller et al. examined disengagement from all public GitHub activities using open-ended survey questions, our study focuses specifically on contributors who disengaged from certain scientific OSS communities, allowing for a more targeted analysis of disengagement factors in this domain. Our objective is to identify the pain points in the context of scientific OSS that may not be adequately tackled by current solutions.

III. Research Methods

Given the diverse scientific domains within scientific OSS communities, we focus our study on a single scientific domain—astronomy, using the *Astropy* project—to minimize domain-specific biases and ensure a more consistent analysis. To achieve broader demographic coverage, we employ a survey-based approach targeting disengaged contributors with open-ended questions, enabling us to reach a large and representative sample to address our research questions while also capturing the nuanced perspectives of participants. In this section, we first justify our choice of study subject, followed by the introduction to the survey design and analysis method.

A. *Study Subject – the Astropy Project*

We selected the *Astropy* project [77], a well-established OSS ecosystem comprising 51 software packages in astronomy, as our study subject. Over the past decade, *Astropy* community-published papers have received over 18,000 citations [50]–[52], and the core repository, *astropy/astropy*, has over 1,800 forks on GitHub. Widely used in both astronomy research and scientific missions [78], [79], the *Astropy* ecosystem offers publicly accessible code repositories and artifacts, providing a rich context to examine tangible issues in community sustainability. With over 10 years of commits history and over 1,000 commit contributors among the software packages in the ecosystem, it provides sufficient data to investigate the sustainability issues within the scientific OSS communities.

B. *Survey with Disengaged Contributors*

Recruitment. We mined and analyzed the commits history (til Aug. 2022) across repositories within the *Astropy* ecosystem to identify contributors who have disengaged. Our analysis focuses on "contributors," defined as individuals who made codebase commits. We distinguish them from "non-code contributors," such as those engaging in issue tracking or online discussions, which are challenging to detect accurately at scale and track over time. Disengaged contributors are defined as those who meet all of the following criteria (**C1-3**):

- **C1:** The contributor has merged at least one commit to any of the 51 packages before Aug. 2022, and
- **C2:** The contributor has not made any contributions to the code base within the last 100 days or more, and
- **C3:** The contributor is still active in other OSS projects outside of the *Astropy* project ecosystem.

This activity pattern serves as an indicator of having recently ceased or departed from participation in the *Astropy* community. Note that related work has explored different thresholds (e.g., 30, 90, 180 days) [54], [80] and showed similar results.

The detailed procedure can be divided into three steps as Figure 1 demonstrates:

Fig. 1. Steps to select survey participants.

- *Step 1*: We analyzed the commit history of all 51 packages to identify commit authors, resulting in 1,116 GitHub accounts.
- *Step 2*: For each account, we collected their latest development activity (i.e., commits, PRs, and issues) via GitHub API and identified the potential disengaged contributors using the criteria C1-3 described above. We further removed bot accounts and deleted accounts. This results in 469 GitHub accounts.
- *Step 3*: We further filtered out the ones without public contact information [81], resulting in 292 potential participants to whom we sent out survey invitations.

Survey Design and Analysis. We ask the following three open-ended questions [82] in our survey:

1) **Motivations for contributing to the software project in *Astropy* ecosystem:** *What were your motivations for contributing to [package name in Astropy ecosystem]?*
2) **Reason for disengaging from the community:** *Why did you stop contributing after [contribution date]?*
3) **Suggestions for retaining the contributors and sustaining the *Astropy* community and the scientific OSS community in general:** *Is there anything the scientific open-source software community can do to keep contributors stay longer in the community?*

In total, we received *80* responses (response rate 23%). To identify the salient themes related to community sustainability challenges from the perspective of disengaged contributors, We used a combination of deductive and inductive coding [83] to analyze survey responses. For deductive coding, we focused on motivations for contribution and reasons for disengagement, applying codebooks from prior work by Gerosa et al. [62] and Miller et al. [63]. Two authors independently coded 16 responses (20%) out of 80 using these codebooks. For inductive coding, we examined the text responses to identify new themes, specifically extracting suggestions from contributors on sustaining community engagement. To ensure coding consistency, we applied the *constant comparison method* [84], systematically comparing new codes with existing ones to refine and validate our categories. In cases of disagreement, we iteratively compared and discussed interpretations until reaching consensus. Through this process, we established a final set of agreed-upon codes. One author then applied this refined coding scheme to the remaining data, allowing us to capture both predefined and new themes effectively. All data were anonymized in our reporting by indexing each response as *Survey Participant id* (SP[id]). This study received approval from the Research Ethics Board at the authors' institutions. For coded results and survey questions, please refer to [85].

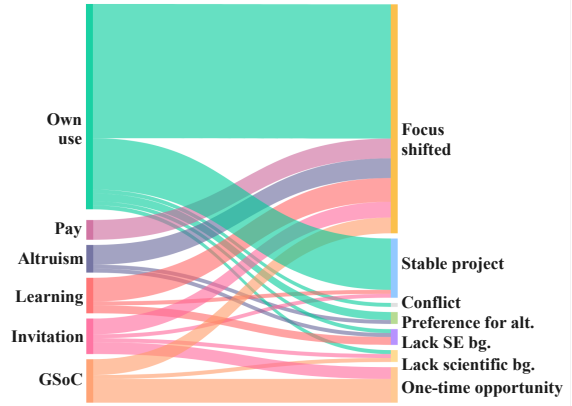

Fig. 2. Mapping between motivations and reasons of disengagement.

IV. Result: (RQ1&2) Motivation & Disengagement

A. *Motivation for Contributing to Communities in* Astropy

From analyzing the survey responses with established codes from Gerosa et al. [62], we identified six types of motivations, fewer than the 10 motivations found in the original study. This difference may be due to the smaller sample size in our survey. As no new themes emerged, we adopted the terminology used by [62] to describe the motivations identified for contribution in the scientific OSS context (see Table I).

In contrast to Gerosa et al's findings [62] which showed a recent decrease in *own-use* as motivation for OSS development, our results reveal that it remains the dominant motivation for contributing to *Astropy*. Among all the *80* participants, 51 contributed to *Astropy* during their *research/education/hobby-related work* in the science domain. Their contributions include fixing bugs or enhancing existing features for *Astropy* as needed for their own research projects. These code changes in turn benefit the community at large. Most of the cases are self-motivated contributions with some exceptions; SP19's contribution was by *invitation* – "*I had found an issue in the software and was asked to contribute a solution to it (if I was able to) by the project maintainers*."

Finding 1: To compare our replication study with prior research, for **motivations to contribute**, the primary reason in general OSS communities has shifted from *own-use* to *altruism* and *learning* over the past decade [62]. However, our survey results reveal that *own-use* remains the primary motivation in scientific OSS, likely due to its predominant role in research settings.

B. *Reasons for Contributor Disengagement*

We identified three primary factors–*lack of motivations*, *high entry barriers*, and *conflicts* (see Table II)–that align with the findings from Miller et al.'s work [63].

A significant portion of survey participants cease their contributions due to ***lack of motivations***. Specifically, 52 out of *80* survey participants stopped contributing because their *work focus had shifted*, such as career changes, graduation, or moving away from academia. Participants recognized this

Themes	Description	Example response	#
Own-use	During research/education/hobby usage	"*...those functions were essential for my research, I developed&committed onto* Astropy." (SP63)	52
Altrusim	Benefit others via OSS	"*Sharing my work so others can benefit thereby.*" (SP61)	8
Learning	Gain experience via OSS	"*Learn how to structure my programming better.*" (SP66)	8
Invitation	Via OSS Meet-up, or research workshop	"*It was part of a Hacktoberfest event and I knew someone involved in the project.*" (SP11)	6
Pay	As SDE or researcher	"*I was employed as a programmer in a lab.*" (SP4)	5
GSoC	Google Summer of Code	"*I was looking for a summer intern, so I started contributing to* Astropy *for GSoC.*" (SP55)	8

Table I: Motivation for contributing to *Astropy* ecosystem. Note that the listed motivations are not mutually exclusive. One survey respondent can mention multiple motivations for contribution and would be counted for each category.

disengagement as inevitable, given that their primary motivation was research-focused *own-use*. SP15 noted, "*I have since moved on to another area of physics. The question of how to keep contributors engaged in coding for [this software] (or similar scientific software) is essentially tied to how to retain researchers within their fields*." Similarly, contributors whose involvement was driven by a specific research need felt no ongoing motivation to remain engaged. SP16 explained, "*I did a pretty minor one-off bug fix that I noticed. I wasn't planning on working on a longer-term development project*." Furthermore, 15 of the *80* participants reported that they are still open to contributing, but the current *project is stable* and already meets their needs. Additionally, eight survey participants contributed to *Astropy* via *Google Summer of Code (GSoC)*, which is an annual program that pays university students stipends to develop features for various OSS projects [6], [86]. Two survey participants only contributed one PR each in order to apply for GSoC internship (as part of the application requirement), and they did not continue contributing since their applications were rejected. While the other six interns discontinued their participation after the internship, mainly due to shifts in their focus. This observation aligns with the observation made by [6] who studied community engagement via GSoC and found that it is not a reliable source.

A further notable reason for disengagement is the ***high entry barrier*** for both scientific and software engineering knowledge. three participants explained that without enough SE knowledge, it is difficult to use GitHub and make contributions that meet the requirements defined by the maintenance team. Additionally, three participants cited the steep learning curve associated with the domain-specific knowledge underlying the complex codebase of the packages.

Reasons	Description	#
Lack of motivations	Focus shifted	53
	One-time opportunity	9
	Project is stable	15
High entry barrier	For science outsider	3
	For scientists who lack SE background	3
Preference for alternative	Found alternative project with similar functionalities that better aligns with contributors' requirements	2
Conflicts	Conflicts with team members on project plan	1

Table II: Reasons for disengagement in *Astropy* ecosystem.

Beyond *lack of motivation* and *high entry barriers*, two contributors disengaged after discovering *preferable alternative projects* with similar functionalities: SP3 preferred an alternative project due to its superior performance from being implemented in C/C++. Likewise, SP26 opted for a more flexible but complex alternative project that was better suited for integrating new features and had more active maintenance. These cases indicate that contributors may disengage from a project due to preferences for performance or flexibility considerations when alternative options are available.

In some instances, contributors disengaged due to ***conflicts*** regarding project plans and visions. For example, SP34, an initial co-founder, withdrew after realizing that their vision for a "*numerical relativity-based Python package*" did not align with those of the other core team members: "*I wanted to focus on real big problems like simulating a binary black hole system, and others were interested in smaller things like symbolic calculations.*" Such cases suggest that community fragmentation can occur when differing visions or priorities arise among contributors.

Comparing our findings with Miller et al.'s study, where *occupational reasons*, *social reasons*, and *technical reasons* were cited as common disengagement factors [63], we find both similarities and unique aspects within the scientific OSS context. Occupational reasons, such as shifts in research focus or career changes, were also prominent in our study, with contributors often leaving due to academic transitions or changes in scientific interests. However, the *high entry barriers* in scientific OSS—requiring both specialized domain knowledge and software engineering skills—introduce challenges that may be less pronounced in general OSS. We also identified additional reasons, such as discovering *preferable alternative projects* and *conflicts* over project vision, which add nuance to disengagement reasons beyond those identified by Miller et al. [63].

Mapping between motivations and reason for disengagement. Lastly, we connect each survey participant's motivations and reasons for disengaging. As shown in Fig. 2, a large portion of the contributors with the motivation of *own-use* stopped after their research focus shifted or the project became good enough for their usage. Also, people who made several

commits during a short period of time (e.g., GSoC, during research, or OSS summit) stopped contributing right after the end of the event.

Finding 2: With *own-use* as the primary motivation, the **main reasons for disengagement** in scientific OSS align with findings from prior research: contributors frequently leave due to shifts in work focus or high entry barriers, especially if they lack domain expertise or SE skills. In scientific OSS, this typically involves a shift in research focus or departure from academia. Unlike general OSS, these inherent turnover factors make it challenging to retain contributors through community efforts alone, highlighting the need for alternative sustainability strategies.

V. Result: (RQ3) Suggestions to Improve Sustainability

When sharing thoughts on how to improve sustainability of the community, participants often talk about a challenge and corresponding solution. Overall, as demonstrated in Table III, we identified four key areas of challenge within the scientific OSS community: *onboarding newcomers and retaining contributors*, *engineering work being undervalued*, *obstacles to inclusivity and engagement in community building*, and *redundant development and fragmented communities within the ecosystem*. While some of these challenges are common in broader OSS communities, others are unique to scientific OSS, posing distinct risks to the sustainability of both the software and its community.

A. *Challenge 1: Difficulty in Onboarding Newcomers and Retaining Contributors*

1) *Better support for contribution workflow*

16 survey respondents noted that the existing contribution process can be enhanced, such as with improved mentorship support, making it more welcoming for new contributors, simplifying the setup for the development environment, and faster responses during code reviews.

Challenges	Suggestions	#
Onboarding newcomers & retaining contributors	Better support for contribution workflow	16
	Train scientists on SE best practices	3
	Offer more fellowship/internship opportunities	3
	Lower entry barrier in science	2
	Smooth transition of contributor turnover	1
Undervalued engineering work	Provide financial incentives	12
	Gain recognition through the academic reward system	8
	Acknowledge the impact of contribution	7
	Publish papers for the scientific OSS	1
Building inclusive &engaging communities	Foster a welcoming community culture	7
	Organize gatherings to keep community engaged	4
	Accept contribution at all levels inclusively	2
Fragmented communities	Call for collaboration to reduce duplicated efforts	1

Table III: Identified challenges and corresponding suggestions of improving the sustainability of *Astropy* community

For existing contributors, several contributors highlighted the need for improved contribution workflows, especially in areas outside their expertise, like UI/UX design. SP4 noted, "*What would have helped me the most is as much plug-and-play UI/UX infrastructure for creating public-facing features. I didn't have much experience in UI/UX and was figuring it out as I went. That's super time-consuming and may be one of the reasons why the project didn't go farther than it did. If I only focused on the medical imaging-specific tasks, things would have likely gone a bit faster.*" This illustrates how support for workflow in regarding SE practices could allow contributors with domain knowledge to focus more on scientific tasks. Additionally, contributors pointed out that certain procedural hurdles during PR reviews, such as strict code style or specific unit testing requirements, could deter engagement. SP56 explained, "*Listening to others' experiences with open-source projects, sometimes the PR review procedure can present hurdles which the contributor sees as unnecessary...This can create long drawn-out review processes which the contributor feels are a waste of their time and that their contribution is not of high quality or wanted by the community*". This feedback suggests that regularly reviewing and clarifying PR requirements and procedures from a new contributor's perspective, possibly using past PRs as case studies, could help reduce barriers and make the contribution process more welcoming and efficient. Others also emphasize timely response to issues and PRs can be important for encouraging contribution (SP37, 38).

For onboarding newcomers, participants emphasized the need for clear contribution guidelines and a well-organized "to-do" list (SP36, 44, 56). More importantly, participants mentioned the complexity of setting up the development environment can discourage contributions (SP21, 44). For example, SP44 noted "*I had to setup my local dev environment and it was a bit cumbersome because project had a lot of dependencies with c/c++ so it was difficult to setup (as I was using Windows at that time). In the end, I discard the idea of the contribution*" (SP44), suggesting that tools like Docker or Kubernetes could simplify dependency management and make contributing easier.

2) *Training scientists for SE best practices*

SP10 and SP21 highlight the need for training in SE best practices for scientists with limited formal SE backgrounds. SP21 mentioned the "*my software development and GitHub skills are very poor! I never received any formal training in software development, I find GitHub very non-intuitive, and so I just stayed in my comfort zone as far as coding goes,*" and suggesting "*hand-holding is needed for people like me, but that's not fair to ask of others.*" Participants brought up the need for training in good software development practices, improved software design, and the use of testing to facilitate effective collaboration within the scientific community. This finding is similar to that in prior work on scientific software development [75].

*3) **Offering research fellowship/internship opportunities to attract more newcomers***

Participants emphasized the value of initiatives like GSoC, research meetups, and OSS summits for drawing new contributors. SP24 praised the 2018 Outreachy Summer Fellowship, which supported multiple scientific OSS projects, recommending the continuation of similar programs to strengthen newcomer engagement. However, some participants noted the limitations of initiatives like Google Summer of Code and Hacktoberfest, emphasizing the need for targeted outreach to attract domain experts for sustained contributions. SP78 explained that while these events boost exposure, "*only those who need the project will continue to contribute.*"

*4) **Lowering the entry barrier in science***

Participants reported that contributing to projects in the *Astropy* ecosystem is challenging due to the required domain-specific knowledge. To address this problem, SP41 suggested that "*offering BootCamp sessions explaining the theoretical knowledge behind functions or features to cultivate participant interest in the science domain"* could potentially attract more newcomers joining and retain existing contributors.

*5) **Smoothing transition between contributors turnover***

One can argue that a community should not rely too much on graduate students to contribute to the project. However, survey participant, SP20, who is the founder of one package but left the community due to work shift, said that "*the best solution is to find a way to pair almost abandoned projects with graduate students that can contribute or even take over the project.*" Given that the shift of research focus is common, a smooth transition can be beneficial for scientific software projects to survive in face of inevitable turnover due to scientific software contributors leaving academia or career change.

B. Challenge 2: Engineering Work is Undervalued

As "*lack of motivation*" is highlighted as the main reason for disengagement, survey participants have offered their viewpoints on the ways to motivate contribution.

*1) **Providing financial incentives***

The most straightforward way is to provide financial incentives to people who spend time on engineering-related tasks, especially for working on something not directly related to their research, as mentioned by 12 of our participants. SP57 noted that "*career paths for such people need to be created and funding opportunities that maintain the most important/used/popular packages need to be created.*" This highlights the need for dedicated financial resources and career support to sustain critical scientific OSS projects and retain contributors who may otherwise be limited by traditional academic funding structures.

*2) **Gaining recognition through academic reward system***

A recurring theme among survey participants, particularly those in academia (professors and graduate students), was the need for the academic reward system to better acknowledge contributions to scientific OSS. Eight participants expressed that the current system undervalues such contributions, with SP27 noting that the development of scientific OSS has "*a negative impact on the scientific career.*" Another SP57 observed that– "*the prevailing opinion in the field is that this is engineering with no direct measurable impact on science and that because of that, contributing to developing and maintaining open source packages does not constitute a valid scientific contribution and is therefore out of scope of the expected work that I do. Contributors to these packages would be willing to contribute more often if it actually counted for something, or somehow made their life easier.*"

This suggests that the lack of formal recognition for OSS contributions within academic evaluations discourages participation, particularly for those needing to demonstrate measurable scientific impact, SP62 stated, "*Basically it is seen as a cost to your career to contribute to open source software.*" To address this, institutions and universities could implement policies that acknowledge OSS contributions as valuable academic work, such as offering course credits for student contributors or fellowship opportunities for faculty involved in significant OSS projects. Integrating OSS contributions into academic rewards could foster sustained engagement, aligning the needs of the OSS community with academic incentives. Another solution is to *raise the awareness of OSS* in the scientific communities. According to our study participants, the rate of publishing source code in scientific fields is still low. As SP14 mentioned, "*...if there is more awareness about open source, it gives more credibility to open source and then it can be an incentive as their efforts are better accounted for.*"

*3) **Acknowledging the impact of contribution***

One intriguing suggestion from seven survey participants is to receive recognition for their contributions and to receive feedback regarding the impact of their work. For example, SP13 said that "*If I had more of a sense for the impact of the bug I discovered, how many people use that function or finding code snippets in other open-source projects which use it, that would help make the intangible benefits more tangible.*" In future research, emphasis could be placed on developing methods and metrics to recognize both types of contributions (engineering or scientific) and creating a dashboard to acknowledge contributions of any kind. For instance, the extent to which a fixed bug affects downstream software relying on the package could serve as a meaningful metric for assessing the impact of bug fixes.

*4) **Publishing papers on scientific OSS projects***

As SP61 suggests that, a publication is "*..the only real currency in the current academic paradigm.*" The communities could have co-authored publications on the scientific software to give credits to contributors to encourage more contribution, which aligns with the suggestion of gaining recognition from the academic reward system. In the *Astropy* ecosystem, the project team of contributors co-authored three publications, which introduce both the technical and open-source aspects. Currently, a similar practice has been adopted for software packages such as NumPy [87] and the SciPy [88]) to acknowledge the contribution of engineering work.

It is important to note that the official *Astropy* Team has already taken critical steps in implementing best practices to

enhance sustainability [51]. While the effort has been well-received within the community, feedback from disengaged contributors from the communities in the ecosystem suggests that there is room for further improvement. Future research could compare a larger number of OSS communities who have adopted similar practices and assess the effectiveness and determine whether they have been utilized as intended.

C. Challenge 3: Overcoming Obstacles to Inclusivity and Engagement in Community Building

1) Fostering a welcoming community culture

This is the solution brought up by seven participants, which is a common strategy that can be applied to all types of OSS communities [89]. Two survey respondents (SP24,67) mentioned that an inclusive and welcoming community is important to encourage people to contribute. Such a suggestion is not specific to the scientific domain. As SP67 pointed out, some OSS communities can be unwelcoming and gatekeeping in terms of expertise so one is not able to contribute meaningfully to a project. Such gatekeeping culture could make it difficult to attract long-term contributors. Contributors may have different backgrounds and contributing priorities, and some may eventually get more involved in the project than others, "*..but the only way to find out which are which is to not drive them away in the first place*."

2) Organizing gatherings to keep community engaged

Four survey participants (SP53, 75, 76, 78) emphasized the value of community gatherings in fostering a positive culture and stronger connections among contributors, which can help sustain engagement. As SP75 shared, "*I personally would have felt more attached to the org if I got to meet other members (virtually, if not in person) for informal talks or even knowledge sharing which would have motivated me to keep up and check back every now and then even if I was finding it hard to find time*." This suggestion aligns with existing best practices for building a vibrant open-source community and can be applied to OSS communities in general.

3) Accepting contribution inclusively

While we primarily define community disengagement through a lack of code contributions, two contributors who have stopped making code commits continue to engage in other meaningful ways (SP10, 51). For example, contributors may support projects indirectly by addressing cross-project issues or maintaining connections within the community through non-coding roles. SP51 noted, "*I would like to think that I am still in the community, given that I still pay attention to the development of [project x in Astropy ecosystem] and recommend [project x] to my students. I also know the main developer of [project x] very well,*" indicating their ongoing involvement despite reduced direct contributions.

Additionally, participants suggested that contributions should not be limited to code commits or code reviews. SP10 commented, "*the idea of keeping contributors for longer inside the community is based solely on thinking contributions only possible through commits and reviews. This may not be showing the full picture. I continued contributing to other open-source projects, and I also provided feedback as a user.*"

This perspective aligns with findings from other OSS communities, where non-code contributions are recognized as essential for sustaining the community but are often underappreciated due to the difficulty in tracking and quantifying these activities [90], [91]. Feng [92] highlights the importance of acknowledging "glue work"—critical yet often overlooked non-code contributions—and proposes a dashboard solution for OSS maintainers to improve visibility and tracking of these contributions, thereby supporting contributor retention. Implementing similar tracking and recognition efforts can benefit scientific OSS communities by attracting contributors who, while less focused on coding, bring valuable domain expertise and broaden the scope of engagement in these projects.

D. Challenge 4: Redundant Development and Fragmented Communities within the Ecosystem

Our findings from RQ2 reveal that some contributors disengage after identifying alternative software that better aligns with their requirements (SP3,26). For example, SP26 shifted from using sncosmo [93] to an alternative project with similar functionalities, finding the alternative's active development and flexibility more attractive to contribute, especially as activity in sncosmo declined following the departure of its lead contributor. Nevertheless, both projects remain maintained but at different levels of frequency, suggesting potential opportunities for further collaboration to combine the strengths of both communities.

Participating in broader ecosystems rather than operating within isolated, individual projects can enhance the sustainability of small scientific software communities, particularly in fields with high contributor turnover due to shifting research priorities. For instance, the PyVo [94] project, originally developed as part of the Virtual Observatory (VO) to standardize data access across multiple astronomical archives, enabled scientists to use a unified tool to access various datasets. Later, the original contributor of PyVo (SP29) transitioned their focus away from the project but advocated for integration with the larger Astropy ecosystem. This integration extended PyVo's utility and ensured its long-term viability by facilitating a transfer of stewardship to the Astropy community. Reflecting on this experience, SP29 noted, "*...my new job responsibilities did not afford me time to continue work on PyVo. Fortunately, the PyVo community was growing, and stewardship of the code was transferred into very capable hands within the Astropy community.*" While similar to the story of PyVo, SP64 developed a package for their Ph.D research, later, they mentioned that "*...since I was moving to a different team, there was no one left behind to maintain it. I donated it to the sncosmo GitHub organization in the hopes that someone would pick it up and maintain it in the future.*" These examples indicate that fostering connections with broader communities can be a strategic approach to sustaining scientific OSS by drawing on shared resources, collective expertise, and ongoing involvement across related

projects, this strategy can reduce the project's reliance on individual contributors, thereby enhancing long-term stability. It is not a universal solution, as successful implementation also depends on factors like encouraging user engagement and attracting sufficient attention to ensure that maintenance responsibilities can be transitioned smoothly.

While the increasing trend of researchers open-sourcing their software and libraries is beneficial for the broader community, it has also led to intensified competition for contributor attention. This proliferation of projects can create confusion among users and potential contributors regarding which projects to adopt or support, and conflicting objectives can dilute the focus and resources within the community. As SP33 observed, "*The open-source community is constantly fighting against itself to keep interest in projects. Everyone creates their own shiny new project and then tries to get others to adopt it so that it gains enough support that they can get funding and stick around. Every year at the various software conferences I attend, there's always a new language or data analyzer/visualizer or framework or container management paradigm that people want to make popular, and it splits the community's focus. So what's needed is some way to set the focus of the open-source community.*" SP33 suggested that scientific OSS projects could benefit from more coordinated, collaborative efforts, similar to those used for scientific data collection. For instance, the astronomy community's decadal survey [95] has successfully unified focus around shared research goals; a similar approach could help facilitate collaborative development within the scientific OSS ecosystem. However, implementing such coordinated efforts could increase the management and coordination workload for maintainers who already have limited resources. Future work could explore effective governance structures and automation tools to alleviate this burden, while fostering unified focus and collaboration across projects within the ecosystem. Mechanisms such as shared roadmaps, community-wide surveys, and ecosystem-level governance could enhance cohesion, reduce redundancy, and support sustained contributor engagement in the scientific OSS community.

Finding 3: Contributors highlighted the need for recognition aligned with academic rewards to sustain the community. Quantitative impact measurement could help to validate their work's importance to academic stakeholders. They also called for improved SE infrastructure to support contribution workflows, allowing them to focus on domain-specific contributions and maximize resource use.

VI. Discussion and Implications

Our study reveals that although *Astropy* serves as a successful example of a scientific OSS ecosystem, it can still face challenges that hinder the long-term engagement of contributors and community sustainability. Here, we discuss actionable insights and recommendations for different stakeholders to effectively address these challenges.

A. *Implication for Scientific OSS Practitioners and Community Maintainers*

While scientific OSS faces sustainability challenges similar to generic OSS communities, our investigation revealed that the entry barrier is elevated. This is attributed to the domain-specific knowledge needed during development, necessitating both software engineering expertise and a scientific background for meaningful contributions to the project. Addressing this requires a multi-pronged approach. First, we recommend developing comprehensive documentation that explains software-related information such as code design rationales and workflow best practices can enhance the understanding of relevant concepts and facilitate informed contributions. Additionally, targeted mentorship programs, such as onboarding sessions and domain-specific boot camps, prepared tutorials, workshops, could accelerate the learning for new contributors, making the community more accessible and attractive. Moreover, listing the required expertise (both scientific and software-related aspects), into the "Good First Issues" can help potential contributors better select and succeed in the tasks.

Surveyees highlighted the importance of non-code contributions, such as providing user support, which is often significant yet underrecognized on coding platforms where metrics primarily emphasize commit activity. To address this, established communities like *Astropy*'s core package have implemented mechanisms to formally acknowledge non-code contributions (e.g., creating tutorials and educational programs) on their official website [96]. Such approaches can serve as a model for other scientific OSS communities, helping them leverage domain expertise and foster collaboration more effectively.

Furthermore, within scientific OSS communities, contributions related to engineering are often undervalued, primarily due to the academic evaluation system. To encourage sustained contributions, especially from researchers who may prioritize scientific over engineering contributions, it is vital to create recognition mechanisms that align with both scientific and engineering outputs. For example, fostering a culture of publicly acknowledging high-impact code contributions—through dashboards or contributor recognition initiatives—can help increase visibility and appreciation for such efforts. Future research could also explore developing metrics to quantitatively assess the impact of code contributions, further reinforcing their value within academic and OSS ecosystems.

B. *Implication for SE Researchers and Tool Builders*

How to substantially reduce the workload for both scientists and software engineers remains a central concern for designing tools serving the scientific OSS community. This includes designing workflows that streamline issue tracking, code review, and contributions, allowing scientists to spend more time on domain-specific tasks rather than learning software engineering intricacies. Future studies could explore automated tooling that categorizes issues and offers guidance that includes the essential scientific theory behind the issue and the necessary programming knowledge. Tools to break down the sub-tasks nec-

essary to resolve the related issues would also be beneficial in emulating a step-by-step onboarding process for newcomers.

Furthermore, we suggest that SE researchers investigate quantitative metrics to appropriately acknowledge contributor impact beyond traditional measures, such as download counts and citations of scientific open-source software. To encourage long-term commitment, as highlighted by our survey participants, it is also essential to demonstrate the impact of their contributions qualitatively. By capturing broader dimensions of OSS impact, these metrics can provide deeper insights into the value of engineering contributions in a way that resonates with the academic and scientific communities.

Additionally, SE infrastructure enhancements—such as simplified workflows, improved documentation, and automated testing—could help contributors navigate technical barriers, freeing them to focus on high-value, domain-specific tasks. By addressing these challenges, SE research could contribute significantly to sustaining scientific OSS communities and fostering long-term contributor retention.

With advances in large language models (LLMs), more LLM-based tools have emerged to support SE tasks [97] and scientific research [98], [99]. These tools hold promise for assisting developers and scientists with labor-intensive tasks and helping them acquire new skills. For example, Jimenez et al. [100] introduced a benchmark applying LLMs to resolve real-world GitHub issues, extending beyond simple code generation. While results showed that LLMs struggled with complex issues, they demonstrated potential for simpler ones. Notably, the Astropy package showed a lower issue resolution rate than other projects, suggesting that future research could focus on optimizing LLMs for scientific software development. LLM-based tools have the potential to assist scientists with time-consuming development tasks such as code review, documentation, and tutorial generation by integrating both scientific domain knowledge and SE-specific knowledge of the codebase.

C. *Implication for Funding & Research Institutions*

Scientists hold different mindsets when it comes to task prioritization during software development. It is not surprising that scientists lack the motivation to invest effort into code quality, given that academic reputation is mainly based on scientific contribution [75]. This could lead to rejection and abandoned PRs during the code review process, discouraging scientists from making continuous contributions. To bridge the gap between scientific and engineering motivations, funding agencies and research institutions might consider creating grant mechanisms that reward contributions to community-maintained codebases, particularly those essential to scientific research. Additionally, it is important to provide institutional support to acknowledge the contribution to engineering work alongside the scientific ones, such as creating career paths for scientists in research SE and providing incentives for them to invest more effort in improving software quality. Meanwhile, allocating long-term financial support for maintaining the necessary infrastructure for scientific OSS is also vital.

VII. Threats to Validity

Here, we discuss potential threats to validity that may influence our results and interpretations. First, for **construct validity**, as we define the disengagement based on the code contribution activities, it might not fully capture disengagement, as the contributors could stay involved in the community through non-code contributions or intend to resume contributions later. Further, regarding **internal validity**, we use survey to learn from the experience of disengaged contributors. Survey respondents may be more likely to have strong opinions on disengagement and retention, potentially underrepresenting those with more neutral or indifferent experiences in this study. Future work could broaden the scope by surveying active contributors and conducting interviews to gain deeper and more diverse perspectives on the retention of scientific OSS community. For **external validity**, we choose study subject to be *Astropy* ecosystem, which is a specific scientific OSS ecosystem in astronomy domain, the finding may not generalize to other scientific OSS communities with different structures, scales or domains. Although scientific OSS can share common challenges, the unique community structure scientific domain of the Astropy project could limit the applicability of the findings to general OSS communities or scientific OSS communities in other domains. Future work can look into comparing the findings of this study on Astropy ecosystem to other scientific OSS ecosystems. Finally, as we employed coding methods to analyze survey responses, which may have introduced unintended bias due to interpretation. To minimize this, multiple coders independently reviewed the data and reached a consensus on final codes. However, subjective interpretations of survey responses may still impact consistency.

VIII. Conclusion

In this replication study, we study the sustainability challenges of scientific OSS communities. Specifically, we investigate the motivations for contribution, and reasons for disengagement as well as the improvement suggestions from the practitioners' perspectives. Our results show that there are many unique properties in the scientific domain that differ from general OSS in terms of sustaining community contributions. Specifically, while motivations in general OSS have shifted toward social motivations such as *altruism*, scientific OSS contributors remain primarily motivated by *own-use*, which aligns with the research-oriented nature of these projects. This reliance on *own-use* also explains why contributors often disengage due to shifts in research focus or high entry barriers, indicating that community-driven retention efforts alone may be insufficient. Lastly, we propose future research and tooling directions to address these challenges.

IX. Acknowledgments

We express our gratitude to the Astropy community for sharing their valuable insights and to the anonymous reviewers for their constructive feedback, which helped enhance this paper. This work was partially supported by the Alfred P. Sloan Foundation (G-2022-19472).

References

[1] M. Gruenpeter *et al.*, "Defining Research Software: A controversial discussion," Zenodo, Tech. Rep., Sep. 2021.

[2] "Imagej: open source software for processing and analyzing scientific images," 2024. [Online]. Available: https://imagej.net/

[3] P. J. Cock *et al.*, "Biopython: freely available python tools for computational molecular biology and bioinformatics," *Bioinformatics*, vol. 25, no. 11, p. 1422, 2009.

[4] "Software for gravitational wave data," 2024. [Online]. Available: https://gwosc.org/software/

[5] R. E. Amaro and A. J. Mulholland, "Biomolecular simulations in the time of covid-19, and after," *Computing in science & engineering*, vol. 22, no. 6, pp. 30–36, 2020.

[6] E. H. Trainer *et al.*, "Community code engagements: summer of code & hackathons for community building in scientific software," in *Proceedings of the 18th International Conference on Supporting Group Work*, 2014, pp. 111–121.

[7] A.-L. Lamprecht *et al.*, "Towards fair principles for research software," *Data Science*, vol. 3, no. 1, pp. 37–59, 2020.

[8] D. S. Katz, "Towards sustainable research software," Dec. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5748175

[9] Q. Zhang *et al.*, "Research software current state assessment," Sep. 2021. [Online]. Available: https://alliancecan.ca/sites/default/files/2022-03/RS_Current_State_Report.pdf

[10] "Digital research alliance of canada: Research software," 2024. [Online]. Available: https://alliancecan.ca/en/services/research-software

[11] "Canarie: Research software," 2024. [Online]. Available: https://www.canarie.ca/software/

[12] "Software sustainability institute," 2024. [Online]. Available: https://www.software.ac.uk/about

[13] "European commission: Horizon europe," 2024. [Online]. Available: https://research-and-innovation.ec.europa.eu/funding/funding-opportunities/funding-programmes-and-open-calls/horizon-europe_en

[14] "Everse: European virtual institute for research software excellence," 2024. [Online]. Available: https://everse.software/

[15] "Horizon europe framework programme: Horizon-infra-2023-eosc-01-02, development of community-based approaches for ensuring and improving the quality of scientific software and code," 2024. [Online]. Available: https://ec.europa.eu/info/funding-tenders/opportunities/portal/screen/opportunities/topic-details/horizon-infra-2023-eosc-01-02

[16] Z. Merali, "Computational science:... error." *Nature*, vol. 467, no. 7317, pp. 775–777, 2010.

[17] "The low quality of scientific code," 2014. [Online]. Available: https://techblog.bozho.net/the-astonishingly-low-quality-of-scientific-code/

[18] V. Stodden and S. Miguez, "Best practices for computational science: Software infrastructure and environments for reproducible and extensible research," *Journal of Open Research Software*, 2013.

[19] J. Segal, "Software development cultures and cooperation problems: A field study of the early stages of development of software for a scientific community," *Computer Supported Cooperative Work (CSCW)*, vol. 18, no. 5, pp. 581–606, 2009.

[20] ——, "Scientists and software engineers: a tale of two cultures," p. 8, 2008.

[21] C. Morris and J. Segal, "Some challenges facing scientific software developers: The case of molecular biology," in *2009 Fifth IEEE International Conference on e-Science*. IEEE, 2009, pp. 216–222.

[22] J. C. Carver *et al.*, "Software Development Environments for Scientific and Engineering Software: A Series of Case Studies," in *29th International Conference on Software Engineering (ICSE'07)*. Minneapolis, MN: IEEE, May 2007, pp. 550–559.

[23] T. Storer, "Bridging the Chasm: A Survey of Software Engineering Practice in Scientific Programming," *ACM Computing Surveys*, vol. 50, no. 4, pp. 1–32, Jul. 2018.

[24] D. Paine and C. P. Lee, ""Who Has Plots?": Contextualizing Scientific Software, Practice, and Visualizations," *Proceedings of the ACM on Human-Computer Interaction*, vol. 1, no. CSCW, pp. 1–21, Dec. 2017.

[25] M. A. Heroux *et al.*, "An overview of the Trilinos project," *ACM Transactions on Mathematical Software*, vol. 31, no. 3, pp. 397–423, Sep. 2005.

[26] D. Kelly, "Scientific software development viewed as knowledge acquisition: Towards understanding the development of risk-averse scientific software," *Journal of Systems and Software*, vol. 109, pp. 50–61, Nov. 2015.

[27] C. Hine, "Databases as Scientific Instruments and Their Role in the Ordering of Scientific Work," *Social Studies of Science*, vol. 36, no. 2, pp. 269–298, Apr. 2006. [Online]. Available: http://journals.sagepub.com/doi/10.1177/0306312706054047

[28] D. Ribes and T. A. Finholt, "Planning infrastructure for the long-term: Learning from cases in the natural sciences," in *Proceedings of the Third International Conference on e-Social Science*. Citeseer, 2007.

[29] K. A. Lawrence, "Walking the Tightrope: The Balancing Acts of a Large e-Research Project," *Computer Supported Cooperative Work (CSCW)*, vol. 15, no. 4, pp. 385–411, Aug. 2006.

[30] J. Gamalielsson and B. Lundell, "Sustainability of open source software communities beyond a fork: How and why has the libreoffice project evolved?" *Journal of Systems and Software*, vol. 89, pp. 128–145, 2014.

[31] N. Shimada *et al.*, "Github sponsors: exploring a new way to contribute to open source," in *Proceedings of the 44th International Conference on Software Engineering*, 2022, pp. 1058–1069.

[32] N. Juristo and O. S. Gómez, "Replication of software engineering experiments," in *LASER Summer School on Software Engineering, LASER Summer School on Software Engineering, LASER Summer School on Software Engineering*. Springer, 2012, pp. 60–88.

[33] S. Schmidt, "Shall we really do it again? the powerful concept of replication is neglected in the social sciences." 2016.

[34] "The astropy project," 2024. [Online]. Available: https://www.astropy.org/about.html#about-the-astropy-project

[35] F. Queiroz *et al.*, "Science as a game: conceptual model and application in scientific software design," *International Journal of Design Creativity and Innovation*, pp. 1–25, 2022.

[36] U. Kanewala and J. M. Bieman, "Testing scientific software: A systematic literature review," *Information and Software Technology*, vol. 56, no. 10, pp. 1219–1232, Oct. 2014.

[37] X. Lin *et al.*, "Releasing Scientific Software in GitHub: A Case Study on SWMM2PEST," in *2019 IEEE/ACM 14th International Workshop on Software Engineering for Science (SE4Science)*. Montreal, QC, Canada: IEEE, May 2019, pp. 47–50.

[38] H. Hata *et al.*, "Science-software linkage: the challenges of traceability between scientific knowledge and software artifacts," *arXiv preprint arXiv:2104.05891*, 2021.

[39] S. Wattanakriengkrai *et al.*, "Github repositories with links to academic papers: Public access, traceability, and evolution," *Journal of Systems and Software*, vol. 183, p. 111117, 2022.

[40] D. G. Widder *et al.*, "Barriers to Reproducible Scientific Programming," in *2019 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. Memphis, TN, USA: IEEE, Oct. 2019, pp. 217–221.

[41] M. Krafczyk *et al.*, "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*, 2019, pp. 23–28.

[42] R. R. Downs *et al.*, "Community recommendations for sustainable scientific software," *Journal of Open Research Software*, vol. 3, no. 1, 2015.

[43] D. S. Katz *et al.*, "Report on the second workshop on sustainable software for science: Practice and experiences (wssspe2)," *arXiv preprint arXiv:1507.01715*, 2015.

[44] "Better scientific software (bssw)," 2024, https://bssw.io/.

[45] N. S. Foundation., "The cyberinfrastructure for sustained scientific innovation (cssi)." 2022. [Online]. Available: https://beta.nsf.gov/funding/opportunities/cyberinfrastructure-sustained-scientific-innovation-cssi

[46] "Better software for science." 2022, https://sloan.org/programs/digital-technology/better-software-for-science.

[47] P. Ivie and D. Thain, "Reproducibility in scientific computing," *ACM Computing Surveys (CSUR)*, vol. 51, no. 3, pp. 1–36, 2018.

[48] "ropensci: R packages for the sciences," 2024. [Online]. Available: https://ropensci.org/

[49] C. Boettiger *et al.*, "Building software, building community: lessons from the ropensci project," *Journal of open research software*, vol. 3, no. 1, 2015.

[50] A. M. Price-Whelan *et al.*, "Astropy: A Community Python Package for Astronomy," *Astronomy & Astrophysics*, vol. 558, p. A33, Oct. 2013.

[51] ——, "The astropy project: Sustaining and growing a community-oriented open-source project and the latest major release (v5. 0) of the core package," *The Astrophysical Journal*, vol. 935, no. 2, p. 167, 2022.

[52] ——, "The astropy project: building an open-science project and status of the v2. 0 core package," *The Astronomical Journal*, vol. 156, no. 3, p. 123, 2018.

[53] D. Robinson and J. Hand, "Sustainability in research-driven open source software," 2019.

[54] R. Milewicz *et al.*, "Characterizing the Roles of Contributors in Open-Source Scientific Software Projects," in *Proc. Working Conf. Mining Software Repositories (MSR)*. Montreal, QC, Canada: IEEE, May 2019, pp. 421–432.

[55] Z. Codabux *et al.*, "Technical debt in the peer-review documentation of r packages: A ropensci case study," in *2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR)*. IEEE, 2021, pp. 195–206.

[56] I. Chengalur-Smith *et al.*, "Sustainability of free/libre open source projects: A longitudinal study," *Journal of the Association for Information Systems*, vol. 11, no. 11, p. 5, 2010.

[57] R. Krishnamurthy *et al.*, "Peripheral developer participation in open source projects: An empirical analysis," *ACM Transactions on Management Information Systems (TMIS)*, vol. 6, no. 4, pp. 1–31, 2016.

[58] S. O. Alexander Hars, "Working for free? motivations for participating in open-source projects," *International journal of electronic commerce*, vol. 6, no. 3, pp. 25–39, 2002.

[59] K. R. Lakhani and R. G. Wolf, "Why hackers do what they do: Understanding motivation and effort in free/open source software projects," *Open Source Software Projects (September 2003)*, 2003.

[60] R. A. Ghosh *et al.*, "Free/libre and open source software: Survey and study," 2002.

[61] G. Von Krogh *et al.*, "Carrots and rainbows: Motivation and social practice in open source software development," *MIS quarterly*, pp. 649–676, 2012.

[62] M. Gerosa *et al.*, "The Shifting Sands of Motivation: Revisiting What Drives Contributors in Open Source," in *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. Madrid, ES: IEEE, May 2021, pp. 1046–1058.

[63] C. Miller *et al.*, "Why do people give up flossing? a study of contributor disengagement in open source," in *IFIP International Conference on Open Source Systems*. Springer, 2019, pp. 116–129.

[64] P. Gray, "To disengage or not to disengage: a look at contributor disengagement in open source software," in *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings*, 2022, pp. 328–330.

[65] C. Bird, "Sociotechnical coordination and collaboration in open source software," in *2011 27th IEEE International Conference on Software Maintenance (ICSM)*. IEEE, 2011, pp. 568–573.

[66] X. Tan *et al.*, "A first look at good first issues on github," in *Proc. Int'l Symposium Foundations of Software Engineering (FSE)*, 2020, pp. 398–409.

[67] F. Fagerholm *et al.*, "The role of mentoring and project characteristics for onboarding in open source software projects," in *Proceedings of the 8th ACM/IEEE international symposium on empirical software engineering and measurement*, 2014, pp. 1–10.

[68] J. Silva *et al.*, "A theory of the engagement in open source projects via summer of code programs," in *Proc. Int'l Symposium Foundations of Software Engineering (FSE)*, 2020, pp. 421–431.

[69] C. Overney *et al.*, "How to not get rich: An empirical study of donations in open source," in *Proceedings of the ACM/IEEE 42nd international conference on software engineering*, 2020, pp. 1209–1221.

[70] V. Singh *et al.*, "Codes of conduct in open source software—for warm and fuzzy feelings or equality in community?" *Software Quality Journal*, pp. 1–40, 2021.

[71] R. E. Kraut and P. Resnick, *Building successful online communities: Evidence-based social design*. Mit Press, 2012.

[72] J. Dominic *et al.*, "Conversational bot for newcomers onboarding to open source projects," in *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, 2020, pp. 46–50.

[73] C. Stanik *et al.*, "A simple nlp-based approach to support onboarding and retention in open source communities," in *Proc. Int'l Conf. Software Maintenance and Evolution (ICSME)*. IEEE, 2018, pp. 172–182.

[74] X. Tan *et al.*, "Scaling open source communities: An empirical study of the linux kernel," in *Proc. Int'l Conf. Software Engineering (ICSE)*. IEEE, 2020, pp. 1222–1234.

[75] J. Howison and J. D. Herbsleb, "Incentives and integration in scientific software production," in *Proceedings of the 2013 conference on Computer supported cooperative work - CSCW '13*. San Antonio, Texas, USA: ACM Press, 2013, p. 459.

[76] M. Germonprez *et al.*, "Scientific open source software: Opportunities to accelerate scientific progress," 2020.

[77] "The Astropy Project," 2024. [Online]. Available: https://www.astropy.org/

[78] "James webb space telescope: Goddard space flight center," 2024. [Online]. Available: https://webb.nasa.gov/

[79] "Case study: First image of a black holecase study: First image of a black hole," 2024. [Online]. Available: https://numpy.org/case-studies/blackhole-image/

[80] B. Lin *et al.*, "Developer turnover in global, industrial open source projects: Insights from applying survival analysis," in *2017 IEEE 12th International Conference on Global Software Engineering (ICGSE)*. IEEE, 2017, pp. 66–75.

[81] "Github user profile," 2024. [Online]. Available: https://docs.github.com/en/rest/users/users?apiVersion=2022-11-28#get-a-user

[82] K.-J. Stol and B. Fitzgerald, "The abc of software engineering research," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 27, no. 3, pp. 1–51, 2018.

[83] J. Saldaña, "The coding manual for qualitative researchers," *The coding manual for qualitative researchers*, pp. 1–440, 2021.

[84] A. Strauss and J. Corbin, "Basics of qualitative research techniques," 1998.

[85] "Replication package," 2024. [Online]. Available: https://zenodo.org/uploads/14088082

[86] "Gsoc," 2024, https://summerofcode.withgoogle.com/.

[87] C. R. Harris *et al.*, "Array programming with numpy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.

[88] P. Virtanen *et al.*, "Scipy 1.0: fundamental algorithms for scientific computing in python," *Nature methods*, vol. 17, no. 3, pp. 261–272, 2020.

[89] H. S. Qiu *et al.*, "The signals that potential contributors look for when choosing open-source projects," in *Proc. Conf. Computer Supported Cooperative Work (CSCW)*, vol. 3. Association for Computing Machinery, 2019, place: New York, NY, USA.

[90] ——, "Gender representation among contributors to open-source infrastructure: an analysis of 20 package manager ecosystems," in *2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*. IEEE, 2023, pp. 180–187.

[91] B. Trinkenreich *et al.*, "Hidden figures: Roles and pathways of successful oss contributors," *Proceedings of the ACM on human-computer interaction*, vol. 4, no. CSCW2, pp. 1–22, 2020.

[92] Z. Feng, "Oss unsung heroes: Crafting productive communities invisibly," in *2023 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, 2023, pp. 302–303.

[93] "sncosmo," 2024. [Online]. Available: https://github.com/sncosmo/sncosmo

[94] "Pyvo," 2024. [Online]. Available: https://github.com/astropy/pyvo

[95] "Nasa: Decadal survey," 2024. [Online]. Available: https://science.nasa.gov/earth-science/decadal-surveys/

[96] "Astropy learn team," 2024. [Online]. Available: https://www.astropy.org/team.html#Learn_team

[97] X. Hou *et al.*, "Large language models for software engineering: A systematic literature review," *ACM Transactions on Software Engineering and Methodology*, 2023.

[98] A. Birhane *et al.*, "Science in the age of large language models," *Nature Reviews Physics*, vol. 5, no. 5, pp. 277–280, 2023.

[99] I. Beltagy *et al.*, "Scibert: A pretrained language model for scientific text," *arXiv preprint arXiv:1903.10676*, 2019.

[100] C. E. Jimenez *et al.*, "Swe-bench: Can language models resolve real-world github issues?" *arXiv preprint arXiv:2310.06770*, 2023.