OPEN ACCESS

The Implications of Current Practices Relating to the Sharing, Reuse, and Citation of Research Software for the Future of Research

연구소프트웨어의 공유, 재사용 및 인용과 관련된 현재 관행의 의미

Hyoungjoo Park (박형주)*
Dietmar Wolfram**

ABSTRACT

The purpose of this research is to explore the phenomenon of the sharing, reuse, and citation of research software. These practices are playing an increasingly important role in scholarly communication. The researchers found that the citation and reuse of research software are currently uncommon or at least not reflected in the Data Citation Index (DCI). Such citation was observed, however, for the newer software in a number of prominent repositories. The repositories Comprehensive R Archive Network (CRAN) and Zenodo received the most formal software citations. The researchers observed both formal and informal forms of citation when researchers reused software. The latter form involves mentioning research software in passing in the main text of articles, while formal citations appear in the references section. In addition, our comparative analysis helps to explain the phenomenon of self-citation of research software.

초 록

이 연구의 목적은 연구소프트웨어의 공유, 재사용, 인용 현황을 분석하는 것이다. 학술커뮤니케이션에서 연구소프트웨어는 최근 들어 더욱 중요한 역할을 하고 있다. 현재 연구소프트웨어의 인용이 일반적인 관행이 아니거나, 적어도 데이터인용색인(DCI)이 연구소프트웨어의 인용과 재사용을 제대로 인덱싱하지 못하는 것으로 관찰되었다. 소프트웨어인용은 주요 레포지토리(prominent repositories)에서 발견되었다. 소프트웨어인용이 많은 레포지토리는 CRAN(Comprehensive R Archive Network)과 Zenodo였다. 연구소프트웨어가 재사용되는 경우, 비공식 소프트웨어인용(informal software citation)과 공식 소프트웨어인용(formal software citation)이 동시에 관찰되었다. 비공식 소프트웨어인용은 연구소프트웨어가 논문의 본문에서는 언급되지만 참고문헌에는 없는 경우였고, 공식 소프트웨어인용은 참고문헌에도 있는 경우였다. 또한, 이 연구의 결과는 연구소프트웨어의 자기 인용(self-citation) 현황을 설명했다.

Keywords: research software, software citation, software sharing, software reuse
연구소프트웨어, 소프트웨어 인용, 소프트웨어 공유, 소프트웨어 재사용

* Assistant Professor, Department of Library and Information Science, Chungnam National University (hyoungjoo.park@cnu.ac.kr) (First Author, Corresponding Author)
** Professor, School of Information Studies, University of Wisconsin - Milwaukee, U.S.A. (dwolfram@uwm.edu) (Second Author)

■ 논문접수일자 : 2021년 11월 15일 ■ 최초심사일자 : 2021년 12월 4일 ■ 게재확정일자 : 2021년 12월 16일
■ 정보관리학회지, 38(4), 65-82, 2021. http://dx.doi.org/10.3743/KOSIM.2021.38.4.065

※ Copyright © 2021 Korean Society for Information Management
This is an Open Access article distributed under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 (https://creativecommons.org/licenses/by-nc-nd/4.0/) which permits use, distribution and reproduction in any medium, provided that the article is properly cited, the use is non-commercial and no modifications or adaptations are made.

1. Introduction

In recent years, the open science movement has highlighted the roles of research software in scholarly communication. The availability of the research software widely used in scientific disciplines (Pan et al., 2015) attests to the growing awareness of the importance of software in scholarly communication. For instance, 63% of respondents to a United States National Postdoctoral Association survey responded that it would not have been practical for them to conduct their research without software (Nangia & Katz, 2017). Researchers in bioinformatics also heavily rely on scientific software to conduct their research (Yang et al., 2018). A possible reason includes that the vast majority of today's research demands computational methods (Goble, 2014).

The citation of research software has attracted the attention of researchers relatively recently (Howison & Bullard, 2016; Li, Lin, & Greenberg, 2016; Park & Wolfram, 2019; Pan et al., 2018). Referencing the original bibliographic literature that describes a piece of software helps to ensure that those who share software receive proper credit (Socias et al., 2015). Software developers may be motivated to share their software with the user community for the purpose of deploying, testing, patching, and developing it further. Despite the apparent importance of sharing as well as receiving credit for doing so, software developers' work is rarely cited. Thus, journal articles, which serve as the primary venue for publishing research findings, have not treated research software as citable, at least not formally in the references section along with the traditional forms of literature on which the authors have drawn. Journals such as Science (American Association for the Advancement of Science, 2016), Nature (Springer Nature, n.d.), and the American Astrophysics Society Journals (American Astronomical Society, 2016) require software sharing and citation. The Research Data Alliance Findable Accessible Interoperable Reusable for Research software (FAIR4RS) Working Group (2021) makes finding, reusing, and citing research software easier. A domain-specific registry such as the Astrophysics Source Code Library, which has made code discoverable since 1999 (Astrophysics Source Code Library, n.d.), allows researchers to find software more easily. Practices in this regard are not uniform, although software journals such as the Journal of Open-Source Software, SoftwareX, Bioinformatics, and Computer Science Communications accept software submissions. Although authors can receive scholarly credit from these software papers, direct citation of research software is rare and inconsistent in current practice. Currently, the citation of shared software remains largely informal in scholarly communication (Park & Wolfram, 2019). For instance, footnotes, in-line text, and links are used as ways to acknowledge software use (Howison & Bullard, 2016). Only 41% of the 135 code sites had citation information of software in any form available in astrophysics (Allen, 2021). As Park and Wolfram (2017) noted, examining disciplinary context is important. The same authors tend to use the same shared data repeatedly (Robinson-García, Jiménez-Contreras, & Torres-Salinas, 2015) and a large portion (84 percent, according to a study

of the Dryad repository) of scientific data citation is self-citation (He & Nahar, 2016). Based on these considerations, we formulated the research questions that guided this study as follows:

- RQ1: What are the characteristics of the sharing of research software documented by the Data Citation Index?
- RQ2. What are the characteristics of the identifiers for research software citation?
- RQ3: How prevalent is the self-citation of software?
- RQ4: How do researchers document their reuse of software through formal and informal citation?

2. Literature Review

Software sharing must be a continuous process because software is often updated, as subsequent versions (e.g., Version 1, Version 2, Version 2.1, and so on) are released to correct bugs and provide more advanced features. This sharing occurs in various ways, such as through repositories (e.g., Zenodo), local teams (e.g., code sharing through a local server used by a laboratory research team), and personal websites. The impediments to the sharing of software include greater cost and complexity compared with the sharing of data (Howison & Herbsleb, 2011). The informal mention of software in passing in the main text of articles—for instance, the R packages referred widely across Public Library of Science (PLoS) articles published in 2015 (Pan, Yan, & Hua, 2016)—remains more frequent. To bring consistency and accountability to the sharing process, the FORCE 11 Software Citation Implementation Working Group (Smith et al., 2016) developed and circulated a set of software citation principles suitable for widespread adoption.

The main idea of software reuse consists of both unaltered and modified use. A major advantage of software reuse is that due to a significant need to reduce the number of bugs during the software development process, software reuse rather than entirely newly developing software code is recommended (Keswani, Joshi, & Jatain, 2014). Successful software reuse includes “the systematic practice of developing software from a stock of building blocks so that similarities in requirements and architecture between applications can be exploited to achieve substantial benefits in productivity, quality, and business performance” (Morisio, Ezran, & Tully, 2002, 341). Large-scale software reuse is supported by smaller-scale software reuse (Henry & Faller, 1995). The current popularity of today's open software repositories has shifted in software developers' search and reuse of software code modules without any systematic reuse method. Software designs are created and modified based on third-party components, which are openly available (Mäkitalo et al., 2020).

Software reuse and scholarly credit have naturally been major concerns for the software engineering community. Reuse can minimize the time and effort required to create software code and contribute to the stability of systems by incorporating previously

tested components into open-source projects. The challenges associated with software reuse may be conceptual (e.g., working out the essential elements of software reuse), cultural (e.g., disincentives against reuse for the members of large development teams), infrastructural (e.g., obsolete supporting software and hardware), managerial (e.g., lack of consensus regarding the application of standards across multiple projects), and technological (e.g., lack of common standards within organizations) (Bassett, 1997).

Metadata is central to the software citation framework and tracking software reuse. However, metadata is a challenge for software citation regarding access, credit, identification, and reuse (Niemeyer, Smith, & Katz, 2016). Although software citation is a relatively recent phenomenon, there has been interest in the application of metadata standards, as mentioned by Li, Lin, and Greenberg (2016). For instance, García et al. (2006) developed a software measurement ontology covering software citation. As a more recent study, Hong (2014) developed a multi-level metadata framework that describes the reusability of software for developers. Codemeta.json (Jones et al., 2017) and CITATION.cff (Druskat et al., 2019) are examples of a metadata schema for software helpful in letting others know how to cite a software code. The Software Application Schema is used to describe software for general purposes. The Software Ontology in biomedical research is an example of discipline-specific metadata used to describe tools, types, versions, and provenance of software (Malone et al., 2014). Li, Lin, and Greenberg examined the descriptive metadata elements of a simulation software popular in material science, Atomic/Molecular Massively Parallel Simulator (LAMMPS), in 400 papers and found that descriptive metadata elements of LAMMPS are inconsistent and incomplete.

Software citation not only awards researchers' scholarly credit for their work but also provides an opportunity for tracking attributions of credit. Thus, when research software is submitted to repositories, it is essential to consider future citation of it and to create enriched metadata that will facilitate its discoverability. For this reason, those who share research software should include with its references to other software or to traditional literature such as journal articles, conference proceedings, and books where appropriate, and the traditional literature should include formal citations to software (again, acknowledgment in the references section of a paper or book as opposed to the main text or acknowledgments sections) where appropriate. Such citation allows for detection by citation-indexing services such as Clarivate Analytics' Data Citation Index (DCI). As things stand, there has been little in the way of formal scholarly rewards for institutions and individuals who collect research software for long-term preservation (Park & Wolfram, 2019). By implementing policies for the formal citation of software, such stakeholders as the governing bodies of institutions and data hosting services can help to ensure that researchers and archivists receive proper scholarly credit for their work.

It was in order to address these issues that Thomson Reuters started the DCI in 2012. The DCI, as part of Thomson Reuters' business, was sold in 2016 and

launched as a company called Clarivate Analytics. The DCI indexes research data and research software worldwide, thus providing a single access point for investigating formal software citation practices. As of September 2021, the DCI held more than 13.4 million records (Clarivate Analytics, 2021). In this index, research data and software are linked to literature articles in the WoS as "Associated Data." The DCI tracks and indexes software like the WoS tracks and indexes journal articles, conference proceedings, and books. Notably, the DCI's records include the formal citation history of research software for the present study.

3. Methodology

For this study, we focused on research software in scholarly communication to the exclusion of proprietary software. As just discussed, we gathered our data from the DCI because its infrastructure contains research software from around the world that Clarivate Analytics has selected and validated. We used the DCI—rather than DataCite.org, another source that indexes data and software citation data—because of the links with the WoS, which DataCite.org does not provide. The dataset represents records on May 11, 2018 from the DCI. The DCI, then, served as a starting point for our retrieval of the records of data related to software citation. In order to reduce the differences across research areas and disciplines, we examined the research areas that had the largest numbers of records of software sharing in the DCI, which were (1) Computer Science, (2) Astronomy & Astrophysics, and (3) Science and Technology Other Topics. Together, these three research areas represented 97.28% of the software sharing documented in the DCI. The DCI categorizes its records by four document types. Those document types are software, dataset, data study, and repositories. Software records were identified based on a designation for the kind of record of the DCI. In the DCI, we restricted the scope of the document type to software. We sorted the records by "most highly cited" in the DCI (rather than by year) because the total numbers of software citations were relatively small. To examine the phenomenon of software citation, we extracted all records categorized as software by the DCI. The total times cited count consisted of all cited counts included the WoS Core Collection, Biosis Citation Index (BCI), Chinese Science Citation Database (CSCD), DCI, Russian Science Citation Index (RSCI), and the Scientific Electronic Library Online Citation Index (SciELO CI).

Before research software can be reused, it must first be discoverable. Metadata fields provide essential access points for the discovery of shared software. We focused on more common metadata helpful in assessing software sharing rather than complete metadata. The fields used were author, data repository, description, identifier, languages, subject, title, type, and year. The definition and scope of these metadata were based on the field tags in records of the DCI (Clarivate Analytics, 2020). To explore the software citation phenomenon regarding identifiers, we examined in detail the use of digital object identifiers (DOIs), universal resource locators

(URLs), researcher IDs, Open Researcher, and Contributor Identifiers (ORCIDs). The definitions and scope of metadata elements such as DOIs, URLs, researcher IDs, and ORCIDs were based on the field tags in records of the DCI provided by Clarivate Analytics. Next, we conducted a descriptive analysis across all shared research software over the 40 years by decade recorded in the DCI. We examined software sharing and formal citation by repositories in the DCI. We then assessed formal self-citation based on the cited literature retrieved from the WoS and the DCI. We then applied a citer-based analysis, using a method similar to that used by Lu, Ajiferuke, & Wolfram. (2014), to examine the various manifestations of self-citation (Park & Wolfram, 2017; Robinson-García, Jiménez-Contreras, & Torres-Salinas, 2016). Citer analysis measures author impact based on the number of citers of a cited work (Ajiferuke, Lu, & Wolfram, 2010). We examined articles that cited software at least once from the All Collections of the WoS to address the issue of self-citation. We collected the bibliographic references for the citing articles for each piece of software by using WoS's “Create citation report” feature. We relied on the “Analyze results” function for the citing articles to identify the citers for each publication. Finally, to demonstrate how researchers document their reuse of research software through formal and informal citation, we manually identified the locations of the citations in the individual articles—that is, whether they occurred in the references, abstract, main text, or supplementary materials sections.

4. Results

We examined the total numbers of published software records in the DCI as an indication of research software sharing. Table 1 summarizes the disciplinary division of these records (RQ1). Computer Science had by far the most records in the DCI (85.2%), followed by Astronomy & Astrophysics (6.34%) and Science and Technology Other Topics (5.73%). Despite representing only a portion of the areas into which human knowledge is classified, these three disciplinary divisions have been the focus of much of the development of open research software and of DCI’s indexing efforts.

Table 2 summarizes the distribution of general (or descriptive) metadata fields that served to describe research software sharing in the DCI. Excepting an identifier, which was never included among the elements in Astronomy & Astrophysics and Science and Technology Other Fields, other metadata elements were widely used. In Computer Science, more than 99% of software sharers provided the information for all of the metadata fields. In Astronomy & Astrophysics, more than 88% of software sharers provided all of the other metadata except, as just noted, an identifier (0%). In Science and Technology Other Topics, while less than one-fifth provided an identifier (18.29%), more than 90% of the software sharers provided all of the other metadata. Identifiers, especially those that are unique and sustainable, such as DOIs, facilitate the citability and traceability of research software, so their complete absence from the metadata associated with Astronomy & Astrophysics

〈Table 1〉 Relative distribution of software sharing records among the top research areas indexed by the DCI

research area	total number of records	percentage of records
Computer Science	20,398	85.212%
Astronomy & Astrophysics	1,517	6.337%
Science and Technology Other Topics	1,372	5.731%
Neurosciences & Neurology	544	2.273%
Mathematics	92	0.384%
Meteorology & Atmospheric Sciences	13	0.054%
Music	1	0.004%
History & Philosophy of Science	1	0.004%
total	23,938	100%

〈Table 2〉 Summary of general metadata field usage for software sharing in the DCI

research area	computer science	astronomy & astrophysics	science and technology other topics
author	98.5%	96.2%	99.00%
data repository	99.99%	100%	100%
description	98.64%	100%	100%
identifier	94.18%	0%	18.29%
languages	100%	100%	100%
subject	99.97%	100%	100%
title	99.99%	100%	100%
type	99.99%	100%	95%
year	99.97%	100%	100%
total	99.03%	88.47%	90.25%

and relative rarity in the metadata associated with Science and Technology Other Topics represent a significant impediment to the implementation of an automated retrieval process for research software in these software environments. The prevalence of software sharing in the three research areas, though, points to future improvements in the awarding of formal scholarly credit to those who share software. The data that we collected did not allow us to assess the quality of the metadata or the administrative methods for them in the context of shared software.

Table 3 displays the results of our comparative analysis of identifiers and software citation in the DCI. Table 3 shows the results after expanding the identifiers in Table 2 to include URLs, researcher IDs, and ORCIDs in addition to the results for the DOIs. Our aim here was to examine in detail the phenomenon of the use of identifiers for research software. The various identifiers included, for example, http://dx.doi.org/10.7916/D8XW4K38 as a DOI, http://ascl.net/1107.004 as a URL, Viglione, Alberto/M-4860-2017 as a researcher ID, and

〈Table 3〉 Comparison of identifiers and software citations in the DCI

research area	items	identifiers			
	count of total # of items	count of URL	count of DOI	count of researcher ID number	count of ORCID
Computer Science	20,216	1,173 (5.8%)	19,040 (94.2%)	79 (0.4%)	75 (0.4%)
Astronomy & Astrophysics	1,492	1,492 (100%)	0 (0%)	20 (1.3%)	13 (0.9%)
Science and Technology Other Topics	13,720	11,211 (81.7%)	2,509 (18.3%)	75 (0.5%)	44 (0.3%)
total	35,428	13,876 (38.83%)	21,549 (60.31%)	174 (0.49%)	132 (0.37%)

〈Table 4〉 DCI-based software citations based on the year of software development (1980-2018)

research area	DCI				
	year	1980-1989	1990-1999	2000-2009	2010-2018
Computer Science	count of total # of items	1	2	485	19,723
	times cited	0	0	0	209
Astronomy & Astrophysics	count of total # of items	0	27	9	1456
	times cited	0	13	4	868
Science and Technology Other Topics	count of total # of items	1	0	1,034	12,685
	times cited	1	0	3	2719
total		3	42	1,531	37,660

Viglione, Alberto/0000-0002-7587-4832 as an ORCID. Disciplinary differences were apparent for software citation with respect to the use of DOIs and URLs. In Computer Science, the vast majority of records (94.2%) used a DOI as the identifier. This result is likely attributable in part to the influence of the repositories in which software associated with Computer Science appeared (as discussed below). By contrast, DOIs were not used as an identifier in Astronomy & Astrophysics and appeared relatively infrequently (in 18.3% of the records) in Science and Technology Other Topics. Conversely, URLs were popular as identifiers in Science and Technology Other Topics (81.7%). Neither researcher ID numbers nor ORCIDs were widely used as identifiers in any of the research areas. As already observed, the use of sustainable and permanent identifiers such as DOIs enables citation tracking and, therefore, facilitates the awarding of recognition to scientific work.

Table 4 displays the citations in the DCI for research software by decade from 1980, the first year in which software was included in the DCI. We

did not include in our count citations that did not provide the publication year (of which there were 3 in Computer Science, 4 each in Astronomy & Astrophysics and Science and Technology Other Topics). The total records of software in the DCI have increased markedly in recent years, specifically since 2010. Considering the total citation count for software, uncitedness was quite frequent, though it varied depending on the research area; thus, 99.70% of the software titles in Computer Science remained uncited, as did 46.85% of the titles in Astronomy & Astrophysics and 88.60% of those in Science and Technology Other Topics.

Table 5 summarizes the levels of software sharing and formal software citation across the repositories in the DCI. For research software to be traceable and sustainable, the infrastructure of software repositories must be designed in light of appropriate citation guidelines given the dynamic and evolving nature of software and the dependency on other libraries or software for traceability and sustainability. We identified 11 major repositories, including both third-party repositories and institutional repositories (e.g., Zenodo and the Columbia University Academic Commons, respectively). A digital repository such as Zenodo enables researchers to designate software code persistently as an identifier rather than the object to be referred to, such as a version number. Based on the assignment of records from each repository to only one research area, it is clear that the categorization of software sharing records takes place at the repository level rather than at the level of individual pieces of software. Therefore, the categorization of research areas may not describe accurately the function of a specific shared software file. Zenodo was the only repository that had received at least one software citation in Computer Science. Compared with a general-purpose repository like Zenodo or Figshare, a discipline-specific repository, such as GenBank or Pangaea, may not indicate when software citation increases. Institution-specific repositories, such as Edinburgh DataShare, ETH Data Archive, and Rutgers University Community Repository tended to engage in software sharing. Software (whether executable or in the form of source code) was preserved in general-purpose digital repositories rather than in discipline-specific digital repositories. The fact that the sciences are based on collaboration involving multiple research teams, disciplines, institutions, and even countries may help to explain this finding. All of the records in Zenodo that we examined had DOIs (e.g., http://dx.doi.org/10.5281/zenodo.10416) rather than URLs as identifiers, reinforcing the notion that persistent identifiers play a vital role in formal software citation. None of the shared software in Computer Science provided software version numbers. For records categorized as Computer Science, software was mostly shared through Zenodo, followed by Figshare and ModelDB. In Astronomy & Astrophysics, software was only shared through the Astrophysics Source Code Library. In Science and Technology Other Topics, Columbia University Academic Commons was the repository through which software was most widely shared, followed by the Comprehensive R Archive Network (CRAN). Only a few repositories indexed by the DCI were being used to share software.

〈Table 5〉 Analysis of software sharing and formal citation by repositories in the DCI

repository	computer science		astronomy & astrophysics		science and technology other topics	
	total # of items	sum of total times cited	total # of items	sum of total times cited	total # of items	sum of total times cited
Astrophysics Source Code Library	0	0	1,492	885	22	0
Columbia University Academic Commons	0	0	0	0	11,149	3
Comprehensive R Archive Network	0	0	0	0	2,547	2,703
Edinburgh DataShare	19	0	0	0	0	0
ETH Data Archive	80	0	0	0	0	0
Figshare	1,589	4	0	0	0	0
ModelDB	1,156	0	0	0	0	0
nanoHUB	0	0	0	0	1	16
Queens Research Portal	3	0	0	0	0	0
Rutgers University Community Repository	1	0	0	0	0	0
Scholars Bank	0	0	0	0	1	0
Zenodo	17,368	205	0	0	0	0
grand total	20,216	209	1,492	885	13,720	2,722

The main repositories in use differed across the research areas. The software available in five of the institutional repositories had received no citations at all. Interestingly, software shared through CRAN received the most formal citations (2,547 shared software records and 2,703 total citations). This finding suggests to us those sharers of developed R packages need to consider using CRAN as a means to receive formal scholarly credit. CRAN has more records available for citation and attracts more citations per record than institutional repositories.

Figure 1 displays a screenshot showing the research software that received the most formal software citations (327 in the DCI) in the research area of Science and Technology Other Topics. Though the link was not in the form of a permanent identifier, such as a DOI, but rather that of a URL (https://cran.r-project.org/web/packages/vegan/index.html), the sharers provided detailed descriptions of the research software in CRAN. The relationship between the high rate of software citation and the detailed description of software is consistent with the findings for data citation (Piwowar, Day, & Fridsma, 2007).

Table 6 summarizes our citer-based analysis of self-citation comparing software in the DCI with the bibliographic level in the WoS's All Databases except the DCI (RQ3). The definition of self-citation extends "to include citations originating from publications authored by one of the coauthors of the cited publication of interest, or coauthor self-citations"

〈Figure 1〉 Screenshot showing the research software that received the most citations in the DCI in the area of Science and Technology Other Topics

〈Table 6〉 Comparisons of self-citation between the software level in the DCI and the bibliographic level in the WoS All Databases (excluding DCI) by citer-based analysis

research area	software level (i.e., shared software)		bibliographic level (i.e., citing article)	
	DCI		WoS All Databases (excluding DCI)	
	total citations with self-citations	total citations without self-citations	total citations with self-citations	total citations without self-citations
Computer Science	6 (0%)	6	82 (1.2%)	81
Astronomy & Astrophysics	1 (0%)	1	85 (0%)	85
Science and Technology Other Topics	1,250 (1.3%)	1,234	221 (0.9%)	219

(Ajiferuke, Lu, & Wolfram, 2010, 2089). Citer-based analysis identifies the origins of citations apart from self-citations. We found that the phenomenon of self-citation, including by co-authors, manifested in distinct ways depending on the research area. In Computer Science and Astronomy & Astrophysics, we observed neither self-citation of software nor bibliographic self-citation (0%). Elsewhere, software self-citation was more common in Science and Technology Other Topics, while bibliographic self-citation was more common in Computer Science. In the former research area, the average percentage of self-citation at the software level (1.3%) was greater than the average percentage at the bibliographic level (0.9%), implying that the same software was cited more than once because of self-citation.

Table 7 displays examples of formal and informal software citation in citing articles retrieved from the WoS and Google Scholar (RQ4). Both formal and informal citation appeared for the sharing, reuse, and citation of research software in current practice. For instance, we found informal software citation that displayed how software citation is referenced in specific sections, such as the abstract, main text and software availability.

5. Discussion

In the current research environment, it is difficult to imagine conducting research without the use of some sort of software (Hannay et al., 2009; Nangia & Katz, 2017; Yang et al., 2018), thus, as software is critical to research success. However, researchers have few incentives for taking on the unpaid work of development and maintenance of research software (Bietz, Baumer, & Lee, 2010). This situation makes it difficult to realize the full potential of software to advance research (Howison & Bullard, 2016). In this regard, receiving formal scholarly credit from their shared software is important for researchers. Implementing formal software citation demands new metadata formats and indexing services of the software such as the DCI. Howison and Herbselb (2011) found

〈Table 7〉 Examples of formal and informal software citation in citing articles for software sharing and reuse from WoS and Google Scholar

type of citations	type of sharing/ reuse	location of the text	sample text
formal software citation	software reuse	main text (methods section)	All analyses were conducted in **R version 3.1.3 (R Development Core Team, 2016)** using the **nlme package (Pinheiro et al., 2017)** to fit GLMMs, **lsmeans package (ength, 2016)** for testing multiple comparisons, and **PMCMR package (Pohlert, 2014)** for non-parametric analyses.
		references	Oksanen, J., F. G. Blanchet, R. Kindt, P. Legendre, P. R. Minchin, R. B. O'Hara, G. L. Simpson, P. Solymos, M. H. H. Stevens, and H. Wagner. 2016. Vegan: Community ecology package [R package version 2.3-3]. http://CRAN.R-project.org/package=vegan. Accessed 13 Feb 2016.
informal software citation	software sharing	abstract	The source code, tutorial and artificial bisulfite datasets are **available at http://bioinfo2.ugr.es/MethylExtract/ and http://sourceforge.net/projects/methylextract/**, and also permanently **accessible from 10.5281/zenodo.7144.**
		main text (conclusion section)	Since the protein functional prediction problem is so close to 'language translation', translation based protein functional prediction is likely to be the most promising approach. Source codes are permanently **accessible from 10.5281/zenodo.7506.**
		software availability (supplementary information)	Illuminaio is an R package available from the Bioconductor project (http://www.bioconductor.org) and from 10.5281/zenodo.7588.

that software developers may want their bibliographies (e.g., software papers) to be cited rather than the software itself because it can help them award grants or apply for tenure. The coordinated efforts of various stakeholders worldwide including researchers, their institutions, funding agencies, and the agencies and individuals responsible for maintaining the infrastructure are needed, to impose standards and integrate new practices (Kats et al., 2019). Formal software citation can also refer to the literature in which the software first appeared, such as the first journal article that described it, as (Socias et al., 2015) observed. However, the authors of manuscripts that describe research involving software code, shared through a digital repository, may find it difficult to identify the most accurate and recent citation information. Socias and colleagues also point out that, even when such information can be found, the contributors to programs may differ from version to version.

This exploratory study found differences in software sharing based on the repositories used. As Park and Wolfram (2019) discussed, the extent to which individual researchers are familiar with various repositories may influence software sharing. Each repository has its guidelines, which influence software sharing in repositories as well. Zenodo is an example of a repository that creates DOIs when submitting a software to the repository, although not all repositories provide DOIs at the time of publication. This study found that a unique identifier was not widely used in metadata across disciplines. When identifiers were examined in more detail, identifiers other than DOIs such as URLs, researcher IDs, and ORCID were rare. Although this is challenging, accurate software citation demands persistent and sustainable identifiers to represent research software effectively. A reason is the nature of software code where more than one physical file and executable versions may coexist. Also, distinct versions of programs often have different developers, which complicates recognition of due scholarly credit to those software developers (Socias et al., 2015). One possible solution to the problem of permanent identifiers for software is the use of a permanent human identifier such as an ORCID and researcher ID to facilitate the traceability and capability of research software. Other identifiers for software citation, such as Research Resource Identifiers (RRIDs), Astrophysical Source Code Library (ASCL) IDs, and Software Heritage IDs, have the desirable features of permanence and stability.

We found that the practices of metadata usage for software differ by disciplines. Using core descriptive metadata and persistent identifiers are crucial elements of software citation because they are mechanisms used for tracking and indexing software citation (Katz et al., 2021). Katz and colleagues mentioned that formal software citation is challenging in the absence of a persistent identifier and a permanent landing page that provides access to the software. Software sharing differs across disciplines; therefore, each discipline needs to develop citation systems that work well for them. Also, providing software metadata by popular disciplinary registries facilitates discoverability of software. Examples of disciplinary registries of software metadata include Omic Tools (Henry et al., 2014) and bio.tools in the life sciences. Common metadata

is also essential to scale to large repositories if there is a lot of repeated information about software because source code files in the same folder typically share common metadata. As Du et al. (2021) observed, software creators can express their preferences regarding citation requests through such general metadata schemas as CodeMeta by using ecosystem-specific files, such as the R CITATION file. CodeMeta is planned to be used by the SciCodes consortium (2021) of scientific software registries and repositories to enable software citation, dissemination and recognition.

Software reuse is needed to build bigger, more complex, less expensive, and more reliable software that can be delivered on time. Software reuse aids in reducing cost and development risk of a project, thus saving the trouble of developing new components in software. The reused software components are already known and understood which leads to the development of more reliable software. A disadvantage of software reuse is that not all reusable software components have proper documentation attached to them (i.e., lack of information about components), so reusers need to exert additional effort to examine the components to be able to make proper use of them. Software reuse is only recommended when minimum negative impact and positive influence are guaranteed, as in the improved quality of software. To ensure this, reusable components must meet quality criteria for sustainable software. An effort to meet the demand for high-quality software includes the Computational Infrastructure for Geodynamics community, an open-source repository in the earth sciences requiring documentation for all software packages that includes a citation statement (Hwang, Pauloo, & Carlen, 2020).

We note that our research has limitations. First, the DCI does not index all of the repositories that may contain research software, though it is one of the few sources that does include data citations. Second, data and software citation are relatively recent practices, so the answers to the research questions that we posed are continuing to evolve.

6. Conclusions

We found that the frequency of formal software citation varied across disciplines, as did the frequency of self-citation. Only a few repositories have been used for software sharing to date. The software made available in CRAN and Zenodo received the most formal software citations. Institutional repositories received almost no software citation. The software that received the highest numbers of software citation displayed detailed description of software. We were not able to determine which identifiers (e.g., URLs with or without ORCIDs) promoted formal software citation owing to the variation in the use of identifiers across repositories that we observed. All of the records that were formally cited in the research area of Computer Science had DOIs, though probably only because this is a feature of Zenodo, in which repository software classified as Computer Science primarily appears. By contrast, all of the records cited formally in the research area of Science and Technology Other Topics had URLs. Software was, proportionately, self-cited more frequently than bib-

liographic literature in the research area of Science and Technology Other Topics. By contrast, bibliographic literature was more likely to be self-cited than software in Computer Science. In situations in which formal scholarly credit could be assigned, the references section was not used for this purpose but rather such locations within journal articles as the abstract, methods, or supplementary information sections. Thus, both informal and formal software citation for software reuse occur in current practice.

References

Ajiferuke, I., Lu, K., & Wolfram, D. (2010). A comparison of citer and citation-based measure outcomes for multiple disciplines. Journal of the American Society for Information Science and Technology, 61(10), 2086-2096. http://doi.org/10.1002/asi.21383

Allen, A. (2021). Citation method, please? A case study in astrophysics. arXiv. Available: https://arxiv.org/pdf/2111.12574.pdf

American Association for the Advancement of Science (2016). Science journals: editorial policies. Available: https://www.science.org/content/page/sciencejournalseditorialpolicies#research-standards

American Astronomical Society (2016). Policy statement on software. Available: https://journals.aas.org/news/policy-statement-on-software

Astrophysics Source Code Library. [n.d.]. Welcome to the ASCL. Available: https://ascl.net

Bassett, P. G. (1997). Framing Software Reuse: Lessons from the Real World. Upper Saddle River: Yourdon Press.

Bietz, M. J., Baumer, E. P. S., & Lee, C. P. (2020). Synergizing in cyberinfrastructure development. Computer Supported Cooperative Work, 19, 245-281. http://doi.org/10.1007/s10606-010-9114-y

Clarivate Analytics (2020). Data Citation Index help. Available: http://images.webofknowledge.com/WOKRS517B4/help/DRCI/index.html

Clarivate Analytics (2021). Data Citation Index. Available: https://clarivate.com/webofsciencegroup/solutions/webofscience-data-citation-index

Druskat, S., Spaaks, J. H., Chue Hong, N., Haines, R., & Baker, J. (2019). Citation file format (CFF) - specifications. Zenodo. http://doi.org/10.5281/zenodo.3515946

Du, C., Cohoon, J., Lopez, P., & Howison, J. (2021). Softcite dataset: a dataset of software mentions in biomedical and economic research publications. Journal of the Association for Information Science and Technology. http://doi.org/10.1002/asi.24454

FAIR for Research Software Working Group (2021). FAIR for research software (FAIR4RS) WG. Available:

https://www.rd-alliance.org/groups/fair-4-research-software-fair4rs-wg

García, F., Bertoa, M. F., Calero, C., Vallecillo, A., Ruíz, F., Piattini, M., & Genero, M. (2006). Towards a consistent terminology for software measurement. Information and Software Technology, 48(8), 631-644. http://doi.org/10.1016/j.infsof.2005.07.001

Goble, C. (2014). Better software, better research. IEEE Internet Computing, 18(5), 4-8. http://doi.org/10.1109/MIC.2014.88

Hannay, J. E., MacLeod, C., Singer, J., Langtangen, H. P., Pfahl, D., & Wilson, G. (2009). How do scientists develop and use scientific software? 2009 ICSE Workshop on Software Engineering for Computational Science and Engineering. Vancouver: IEEE. http://doi.org/10.1109/SECSE.2009.5069155

He, N. & Nahar, V. (2016). Reuse of scientific data in academic publications: an investigation of Dryad digital repository. Aslib Journal of Information Management, 68(4), 478-494. http://doi.org/10.1108/AJIM-01-2016-0008

Henry, E. & Faller, B. (1995). Large-scale industrial reuse to reduce cost and cycle time. IEEE Software, 12(5), 47-53. http://doi.org/10.1109/52.406756

Henry, V., Bandrowski, A. E., Pepin, A.-S., Gonzalez, B. J., & Desfeux, A. (2014). OMICtools: an informative directory for multi-omic data analysis. Database, 2014, 1-5. http://doi.org/10.1093/database/bau069

Hong, N. C. (2014). Minimal information for reusable scientific software. Proceedings of the 2nd Workshop on Working towards Sustainable Scientific Software. Available: http://www.research.ed.ac.uk/portal/files/16773670/MinimalInfoScientificSoftware.pdf

Howison, J. & Bullard, J. (2016). Software in the scientific literature: problems with seeing, finding, and using software mentioned in the biology literature. Journal of the Association for Information Science and Technology, 67(9), 2137-2155. http://doi.org/10.1002/asi.23538

Howison, J. & Herbsleb, J. D. (2011). Scientific software production: incentives and collaboration. Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work, 513-522. http://doi.org/10.1145/1958824.1958904

Hwang, L., Pauloo, R., & Carlen, J. (2020). Assessing the impact of outreach through software citation for community software in geodynamics. Computing in Science & Engineering, 22(1), 16-25. http://doi.org/10.1109/MCSE.2019.2940221

Jones, M. B., Boettiger, C., Mayes, A. C., Smith, A., Slaughter, P., Niemeyer, K., Gil, Y., Fenner, M., Nowak, K., Hahnel, M., Coy, L., Allen, A., Crosas, M., Sands, A., Chue Hong, N., Cruse, P., Katz, D. S., & Goble, C. (2017). CodeMeta. KNB Data Repository. http://doi.org/10.5063/schema/codemeta-2.0

Kats, D. S., Bouquin, D., Chue Hong, N. P., Hausman, J., Jones, C., Chivvis, D., Clark, T., Crosas, M.,

Druskat, S., Fenner, M., Gillespie, T., Gonzalez-Beltran, A., Gruenpeter, M., Habermann, T., Haines, R., Harrison, M., Henneken, E., Hwang, L., Jones, M. B., Alastair, J., Kelly A. A., Kennedy, D. N., Leinweber, K., Rois, F., Robinson, C. B., Todorov, I., Wu, M., & Zhang, Q. (2019). Software citation implementation challenges. Available: https://arxiv.org/ftp/arxiv/papers/1905/1905.08674.pdf

Keswani, R., Joshi, S., & Jatain, A. (2014). Software reuse in practice. Proceedings of the Fourth International Conference on Advanced Computing & Communication Technologies, 159-162. http://doi.org/10.1109/Acct.2014.57

Li, K., Lin, X., & Greenberg, J. (2016). Software citation, reuse and metadata considerations: an exploratory study examining LAMMPS. Proceedings of the 82nd Annual Meeting of the Association for Information Science and Technology, 1-10. http://doi.org/10.1002/pra2.2016.14505301072

Lu, K., Ajiferuke, I., & Wolfram, D. (2014). Extending citer analysis to journal impact evaluation. Scientometrics, 100(1), 245-260. http://doi.org/10.1007/s11192-014-1274-y

Mäkitalo, N., Taivalsaari, A., Kiviluoto, A., & Capilla, R. (2020). On opportunistic software reuse. Computing, 102, 2385-2408. http://doi.org/10.1007/s00607-020-00833-6.

Malone, J., Brown, A., Lister, A. L., Ison, J. Hull, D., Parkinson, H., & Stevens, R. (2014). The software ontology (SWO): a resource for reproducibility in biomedical data analysis, curation and digital preservation. Journal of Biomedical Semantics, 5(25). http://doi.org/10.1186/2041-1480-5-25

Morisio, M., Ezran, M., & Tully, C. (2002). Success and failure factors in software reuse. IEEE Transaction on Software Engineering, 28(4), 340-357. http://doi.org/10.1109/TSE.2002.995420

Nangia, U. & Katz, D. S. (2017). Track 1 paper: surveying the U.S. national postdoctoral association regarding software use and training in research. Workshop on Sustainable Software for Science: Practice and Experiences. http://doi.org/10.5281/zenodo.814220

Pan, X., Yan, E., & Hua, W. (2016). Disciplinary differences of software use and impact in scientific literature. Scientometrics, 109, 1593-1610. http://doi.org/s11192-016-2138-4

Pan, X., Yan, E., Cui, M., & Hua, W. (2018). Examining the usage, citation, and diffusion patterns of bibliometric mapping software: a comparative study of three tools. Journal of Informetrics, 12(2), 481-493. http://doi.org/10.1016/j.joi.2018.03.005

Pan, X., Yan, E., Wang, Q., & Hua, W. (2015). Assessing the impact of software on science: a bootstrapped learning of software entities in full-text papers. Journal of Informetrics, 9(4), 860-871. http://doi.org/10.1016/j.joi.2015.07.012

Park, H. & Wolfram, D. (2017). An examination of research data sharing and re-use: implications for data citation practice. Scientometrics, 111(1), 443-461. http://doi.org/10.1007/s11192-017-2240-2

Park, H. & Wolfram, D. (2019). Research software citation in the Data Citation Index: current practices

and implications for research software sharing and reuse. Journal of Informetrics, 13, 574-582. http://doi.org/10.1016/j.joi.2019.03.005

Piwowar, H. A., Day, R. S., & Fridsma, D. B. (2007). Sharing detailed research data is associated with increased citation rate. PLoS ONE, 2(3), e308. http://doi.org/10.1371/journal.pone.0000308

Robinson-García, N., Jiménez-Contreras, E., & Torres-Salinas, D. (2015). Analyzing data citation practices using the data citation index. Journal of the Association for Information Science and Technology, 67(12), 2964-2975. http://doi.org/10.1002/asi.23529

SciCodes Consortium (2021). SCICODES: consortium of scientific registries and repositories. Available: https://scicodes.net/2021/04/09/welcome-to-scicodes

Smith, A. M., Katz, D. S., Niemeyer, K. E., & FORCE11 Software Citation Working Group. (2016). Software citation principles. PeerJ Computer Science, 2(e86). http://doi.org/10.7717/peerj-cs.86

Socias, S. M., Morin, A., Tomony, M. A., & Sliz, P. (2015). AppCiter: a web application for increasing rates and accuracy of scientific software citation. Structure, 23(5), 807-808. http://doi.org/10.1016/j.str.2015.04.005

Springer Nature. [n.d.]. Reporting standards and availability of data, materials, code and protocols. Available: https://www.nature.com/nature-portfolio/editorial-policies/reporting-standards

Yang, B., Huang, S., Wang, X., & Rousseau, R. (2018). How important is scientific software in bioinformatics research? A comparative study between international and Chinese research communities. Journal of the Association for Information Science and Technology. http://doi.org/10.1002/asi.24031