NIH Request for Information: Best Practices for Sharing NIH Supported Research Software

NIH Notice Number: NOT-OD-24-005

Authors: Allen Lee, Alice Allen, Anita Bandrowski, Daniel Garijo, Lorraine Hwang, David Kennedy, Hervé Ménager, Tom Morrell, Bhavesh Patel, SciCodes Consortium

The SciCodes consortium (https://scicodes.net/) represents over 35 research software registries and repositories devoted to helping researchers make their software available and *useful* to their respective communities. We advocate for open source research software, transparent and reproducible science, standardized and interoperable scientific metadata, and providing credit and recognition to those who create, curate, and maintain the computational methods, tools, and cyberinfrastructure integral to modern scientific research. We applaud the NIH's dedication to open science and have several suggested improvements to the NIH software guidelines based on the diverse experience and expertise of our members.

Comments on the current NIH Best Practices for Sharing Research Software

We believe the current FAQ-style NIH Best Practices for Sharing Research Software can be improved and made easier to follow and implement. While a FAQ is useful as discoverable, supplemental information for those with specific question(s), a prioritized checklist with clear and actionable step-by-step instructions for researchers (e.g., the machine learning reproducibility checklist) might be of greater use, with each best practice turned into a declarative statement. For example, the first best practice could be stated as "Make your research software open," with additional guidance and concrete examples as to how this can be done.

Further, the Best Practices do not cover all the requirements of the FAIR Principles for Research Software (https://doi.org/10.15497/RDA00068). A detailed analysis of the current NIH best practices and FAIR4RS principles is available at https://fair-biors.org/docs/crosswalk.

The FAIR Biomedical Research Software (FAIR-BioRS) Guidelines aims to provide a clear and accessible list of specific best practices with concrete examples (https://doi.org/10.1038/s41597-023-02463-x, https://fair-biors.org). We suggest using these as a baseline for the NIH Best Practices for Sharing Research Software, recognizing that the NIH Best Practices may cover a broader range of topics than the FAIR-BioRS guidelines.

The SciCodes consortium also has specific feedback on the following NIH best practices FAQ entries:

1. **Why should I share software and code as "open source" software?**

There are many additional benefits to sharing software that are not described here. Sharing software enables individuals and communities to improve the quality and (re)usability of code. It reduces duplication of effort and supports transparent, higher quality science that democratizes research while improving trust in the software. For an individual researcher, sharing software can also increase visibility and boost citations, since their implementation can be compared to or used by others. Perhaps most importantly however, is the fact that computational methods are *methods*, and like other scientific methods, should be revealed for transparency to support the trustworthiness and reproducibility of the underlying science.

2. **How do I make software source code "open"?**

   We request the addition of *domain registries and repositories* as an explicit resource that can help researchers make their software "open" and publish them according to their own community established norms and best practices. The current guidelines indirectly mention domain specific registries in "provision of additional metadata", but it would be more useful to provide reference to specific resources[1]. Although GitHub, GitLab, and Bitbucket are excellent software development platforms they should not be considered archival repositories for depositing or publishing software as users can delete or change the software within a given repository at any time. Domain-specific registries and repositories, institutional repositories, generalist repositories like Zenodo, and software archival services such as Software Heritage are all designed to preserve and make software FAIR and should be recommended instead (with the sidenote that GitHub repositories are automatically harvested by Software Heritage and can also be explicitly synchronized with Zenodo to publish GitHub releases on Zenodo).

   Including an explicit license for the software should also be part of the definition of making software "open".

3. **Why should I use a license when distributing code?**

   Without a license, software cannot legally be reused and thus distributing the code is essentially rendered meaningless. One can read the code but cannot use it, copy it, or even run it without a license. A license *must* be chosen when publishing code for others to reuse.

4. **How do I choose a license under which to release software developed as part of an NIH award?**

   We recommend the addition of language to follow guidance provided by the funder and/or host institution which may have additional intellectual property and software

[1] e.g., https://github.com/NLeSC/awesome-research-software-registries or https://scicodes.net/participants/

license restrictions. We also recommend that researchers advocate for open licenses at their institution when they develop grant proposals or publish their software.

5. **How can I make my software citable?**

   It's important that the persistent identifier associated with software (e.g., DOI, RRID, etc.), points to a **specific version or release** of that software to be compliant with the FAIR4RS and FORCE11 Software Citation Guidelines. A properly curated CodeMeta codemeta.json file can also assist in making software citable. CodeMeta is an emerging metadata standard for software metadata based on the industry standard schema.org. Adding a codemeta.json file to a source code package, repository, or registry site provides valuable information, including how to cite the software, to those wishing to use, cite, or index the code.

6. **How should I acknowledge NIH as the funder?**

   We recommend that the funder be included in the documentation (e.g. README file or other narrative documentation) and included in the software metadata using a Research Organization Registry identifier (ROR, e.g., https://ror.org/01cwqze88 for NIH). Grant numbers should also be preserved in the software metadata and should be standardized by a registry or repository into a consistent format.

7. **Are there any restrictions I should consider in deciding whether to share the research software I develop?**

   No comment

8. **Can research software I have developed be allowed for use in medical practice or clinical settings?**

   No comment

9. **Do I have to check software developed for security vulnerabilities prior to sharing it?**

   No comment

10. **What metadata should be considered when sharing research software?**

    Metadata should be available in CodeMeta, a standard and machine processable format (https://codemeta.github.io/) or via a registry that provides CodeMeta as a metadata download option. This could be explicitly provided by the software creator(s) or generated by the registry or repository where the software is registered. We recommend that this section add an explicit requirement to include a unique persistent identifier that points to the exact software version being published or shared in the software metadata. A recommended citation or set of citations for the software should also be included.

11. **To what extent should I include documentation for the software?**

    We strongly recommend that narrative documentation be included that describes the intent and purpose of the software and how to run and use the software with acceptable inputs and expected outputs. Publish API documentation as well if the software provides a public API. Consider adopting community-specific standards for narrative documentation that describes the software e.g., the ODD Protocol https://www.jasss.org/23/2/7.html

12. **Does NIH have any requirements or benchmarks for research software quality before releasing it?**

    Research software should be accompanied by a comprehensive test suite. The Journal of Open Source Software has an additional checklist that may be useful https://joss.readthedocs.io/en/latest/review_checklist.html

The remainder of this document includes responses to questions 3, 4, and 7 from the RFI.

3. What existing standards or criteria do you use to evaluate the openness, FAIRness, quality, and/or security of the software you share or reuse?

a. FAIR metadata assessments from FAIR-EU (e.g., https://github.com/fair-software/howfairis)
b. FAIR-BioRS guidelines mentioned above
c. Guidelines as established by the FAIR4RS Principles

4. Describe the collaborative settings in which you develop and share research software. Name communities or organizations, if any, you participate in that are actively promoting or developing software sharing best practices.

SciCodes is a collaborative organization of over 35 registries and repositories that support researchers in sharing research software. SciCodes has developed best practices (https://doi.org/10.7717/peerj-cs.1023) for these software registries and repositories to help them serve as effective stewards of the software resources and metadata entrusted to them.

7. How can NIH support research software communities of practice to better aid development of best practices for sharing and reuse of high-quality research software?

- Support sustainable funding for software development. A majority of research software is created on short-term grants and doesn't have funding for continued maintenance and development. The NIH should consider establishing processes for determining what software should be maintained and at what funding levels. For example, establishing dedicated funding pools to sustain software with demonstrated scientific relevance, adoption and/or usage in a given research community. Funding pools that support more resource-intensive modernization efforts for legacy scientific software would also be useful, as modernization of aging technology stacks typically require intensive software engineering resources over a defined time interval.
- Support continued education and training by promoting and supporting the diverse communities and projects that help researchers to make software sustainable, reusable, and extensible.
- Fund the development of tools that assist in and automate the process of making software FAIR. A large portion of research software is created by scientists who are not formally trained in software development and better tools can reduce the friction of adopting good practices.
- Evaluation of proposed grants should assess if previous research has followed through on commitments to share / reuse research software.
- NIH should encourage journals to require published, citable source code unless there are patents, patient privacy, national security, or other extenuating circumstances that prevent open access to the source code.
- Explicitly prohibit "source code available on request" statements in data management plans and grant proposals. Research has shown that "available upon request" is the least useful way to share code (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9406794/), and that requested software is usually not provided (https://www.bmj.com/content/382/bmj-2023-075767).
- The NIH does not currently mention sharing software on https://sharing.nih.gov/. Incorporating software as an item to be shared would highlight to researchers the necessity of sharing software for transparent and reproducible science.