Do’s and Don’ts for writing impactful papers about scientific software

Joseph D. Romano, PhD
University of Pennsylvania
October 14, 2021
joseph.romano@pennmedicine.upenn.edu
http://jdr.bio
Twitter: @JDRomano2
Ten simple rules for writing a paper about scientific software

Joseph D. Romano and Jason H. Moore

Published: November 12, 2020 • https://doi.org/10.1371/journal.pcbi.1008390
Rules with a ‘*’ := Heavily opinionated!
Rule 1: Make sure your scientific software is *good* scientific software
Rule 1: Make sure your scientific software is *good* scientific software.
Rule 2: Know the right publication venues

- Some journals offer specific publication types for software papers:
  - Bioinformatics - “Application Notes”
  - BMC Bioinformatics - “Software articles”
  - Journal of Open Source Software (JOSS) - all articles
  - PLOS Computational Biology - “Software articles”
- Pay attention to journal requirements!
  - E.g., page/word/figure limits are usually pretty restrictive
  - Some require evaluations using real (not synthetic) data!
Rule 2: Know the right publication venues

- However, it might be better to publish a “traditional” article in a non-computational journal.

- Ask yourself: Who is my intended audience? What kind of impact do I want this software/paper to have?
Rule 3: Publish for users, not developers*

- Remember: source code and code documentation are often meant for developers rather than the average user.

- Software papers are for your scientific audience; i.e., your desired users.

- (Keep this in mind for Rule 6)
Rule 3: Publish for users, not developers*

- Possible issues when publishing using a “conventional” article type:
  - Might need to be creative how you present your methods and results to fit the paper requirements
    - A good strategy is to **present a handful of use-cases in Results**
  - Reviewers might not know how to handle software papers
    - Contact an editor when in doubt!!!
    - Suggest reviewers with computational expertise if you have the option to do so
  - Write your paper at the appropriate level of technical detail for the “typical reader” of your journal
Rule 4: Have a long-term software management plan

COTS and Reusable Software Management Planning: A Template for Life-Cycle Management

William Anderson
Ed Moris
Dennis Smith
Mary Catherine Ward

October 2007

TECHNICAL REPORT
CMU/SEI-2007-TR-011
ESC-TR-2007-011

https://doi.org/10.5281/zenodo.2159713
Rule 4: Have a long-term software management plan

- Who is responsible for maintaining the software in the future?
- What is the cost of keeping the software (and related tools) online? Have you planned for continued funding?
- Who owns the IP behind the software?
- Will updates / bug fixes be provided?
- What will happen if dependencies go offline?
- When and how will you archive the software?
Rule 5: Safeguard against “link rot”

Markwell and Brooks (2006); https://doi.org/10.1002/bmb.2003.494031010165
Rule 5: Safeguard against “link rot”

- The average URL lifespan is 9.3 years (and decreasing)
- Consider non-institutional web pages
  - Affiliations change
  - Changes in organizational structure wreak havoc on websites
- Use Zenodo and/or FigShare to assign permanent DOIs to (versioned!) software and data
- Try to have semantically meaningful URLs
  - E.g., compare:
Rule 6: Know the difference between documentation and research results*

- Don’t put code documentation in a software paper! Documentation changes when software changes!
  - Any time you put code in a paper, you are making a long-term commitment to syntax and semantics

- Only include example code in the paper if it’s absolutely critical. Appendices might be OK, but do this sparingly.

- However, include a prominent link to (versioned!) documentation in the paper

- It’s not a bad idea to have your documentation directly support results presented in the paper
Rule 7: Use modern tooling

- Use a well-maintained, modern programming language
- Publish your software one one or more packaging index
- Distribute as both raw source code and pre-compiled / pre-packaged versions
- Provide detailed documentation (web-based, auto-generated documentation is great!) and instructions/examples
- Provide contribution instructions and ways to report bugs & ask for help
Rule 8: Be consistent

- Across your software ecosystem, try to maintain consistent:
  - Spelling
  - Punctuation
  - Capitalization
  - Logos
- Apply version control, especially to code documentation
- Establish a consistent naming scheme if the software is part of a larger body of computational research
- Don’t force acronyms - keep them simple or avoid them entirely****
Rule 8: Be consistent
Rule 9: Plan for follow-ups

- It’s not just for padding CVs or increasing citation counts!

- Follow-up papers describe major scientific changes to software (e.g., new algorithms) or datasets (e.g., new data types, overhauled database structure, etc.)

- Demonstrate that the authors rigorously follow the scientific process, and that the project is iterative and builds on previous work
Rule 9: Plan for follow-ups
Rule 9: Plan for follow-ups

- **When not** to write a follow-up paper:
  - Bug fixes or minor feature additions/changes
  - Adding/editing/removing relatively few data elements
  - Overhauled website in the absence of major features being added
  - New directions for the software if those directions haven’t yet been implemented
  - New visualizations* or other ease-of-use tools
Rule 10: Prioritize visibility and availability

ComptoxAI: A toolkit for AI research in computational toxicology

Fig. 1. Overview of the graph machine learning approach used in this study. We build a toxicology-focused graph database (named ComptoxAI) using data aggregated from diverse public databases, and extract a subgraph for QSAR analysis containing chemicals, assays, and genes. We then train and evaluate a graph neural network that predicts whether or not a chemical activates specific toxicity-focused assays from the Tox21 database.

*The full graph database for ComptoxAI can be found at https://comptox.ai, and will be described in a separate, upcoming publication.*
Rule 10: Prioritize visibility and availability

- Search engine optimization isn’t just for businesses!
- Links, links, and more links
- Social media (especially Twitter) is a great way to spread the word
Rule 10: Prioritize visibility and availability
Acknowledgements

- Co-author:
  - Jason H. Moore, PhD

- Funding:
  - K99-LM013646 (PI: Romano)
  - R01-AG066833 (PI: Moore)