Promoting analysis reproducibility with accessibility: An example in evolutionary biology

Luna L. Sanchez Reyes*

School of Natural Sciences, University of California, Merced

and

Emily Jane McTavish

School of Natural Sciences, University of California, Merced

February 4, 2022

Abstract

Reproducibility is essential for scientific development. Efforts to increase scientific reproducibility have focused on increasing availability of code and data. However, availability does not imply accessibility, and the latter is infrequently addressed, even if it is key to achieve full workflow automatization and reproducibility. Using an example in evolutionary biology, we identify factors that have specifically affected accessibility in the natural sciences, and ways researchers can address this to ensure reproducible and automatic workflows. The Open Tree of Life project (OpenTree) has developed a platform that facilitates availability of results from evolutionary biology research. However, baseline computational knowledge and skills required to access scientific results are often not found among target users. While documentation is available, it is often written using highly specialized language that is also inaccessible for the average target user. We identified a set of principles to improve accessibility of OpenTree scientific data, implement them in a tutorial, and elaborate on their general application to code and documentation of scientific workflows.

Keywords: open science, education, R, phylogenetics, tutorials

Introduction

Reproducibility –the extent to which consistent results are obtained when a scientific research experiment is repeated (Curating for Reproducibility Consortium 2021)– is a key aspect of the advancement of science, as it constitutes a minimum standard that allows understanding research products (e.g., methods, data, analysis, results, etc.), to determine their reliability and generality, and eventually build up scientific knowledge and applications based on those products (King 1995, Peng 2011, Powers & Hampton 2019). In the natural sciences, rates of reproducibility are low (Ioannidis 2005, Prinz et al. 2011), which has elicited concerns about a reproducibility crisis in the field (Baker 2016). In response, the scientific community has been developing new principles and standards to incentivize cultural changes in an effort to improve long term reproducibility rates in the natural sciences (Peng 2015, Wilkinson et al. 2016). A standard for reproducibility that has received much attention is scientific availability, which we define as the property of research products to be acquired, copied, analyzed, processed and/or reused, at no financial, legal or technical cost (Arnold et al. 2019), and with no geographic, demographic or temporal barriers for the population (Fecher & Friesike 2014). In this paper, we argue that availability can not be fully achieved without accessibility (Box 1). We identify factors that have particularly affected accessibility in the natural sciences, and ways scientific developers can address them to ensure reproducible and automatic scientific workflows. This principles can be generalized and integrated into a reproducibility curriculum to hopefully provide present learners and future researchers with the tools for successful scientific reproducibility.

Box 1. The role of accessibility to achieve full availability. Availability is generally defined as the ability or potential of something to be used and accessed (Cambridge Dictionary, 2022). Following this definition, accessibility is generally considered a synonym of availability. Yet, in practice, availability does not imply accessibility. One example we find useful to practically differentiate the concepts of availability and accessibility is the "marshmallows in an office" story, which goes like this. A couple of brand new marshmallow bags remained in a common office area after a social gathering. The marshmallows were not claimed by anybody, so they were left in the common are, freely available to be taken or eaten by anyone in the office. Yet, the marshmallow bags stayed in the common area, unopened for days and even weeks. It was not until someone decided to open the bags and placed the marshmallows on a tray, that people started actually eating them. They were gone in a matter of hours. Scientific products are not sweet as marshmallows, but they are a coveted resource. While sharing, describing and documenting makes scientific products available, it does not necessarily makes them accessible. For example, documentation of an available scientific workflow might be written using expert language that is too foreign for a general audience to successfully reproduce the workflow. While some individuals might be able to invest the time to learn the expert language to successfully reproduce the workflow, this will not be the case for the majority, even if it is something that would be useful long term, the short term time investment is often too high for individuals to engage with it, making the workflow and its results effectively inaccessible and hence unavailable to the majority of the community.

To elaborate on our thesis, we list a widely used scientific workflow in phylogenetics, a research field within evolutionary biology that focuses on understanding the shared ancestry of living organisms through the reconstruction of phylogenetic trees from biological data. Phylogenies provide the basis to study and understand all biological processes in an evolutionary context (Dobzhansky 1973). Given the importance of phylogenies, improving reproducibility rates in phylogenetic research is increasingly relevant for many aspects of the natural sciences. The Open Tree of Life project (OpenTree) has developed a computational platform that provides availability of results from phylogenetic research, by creating a standardized database of phylogenetic data that is used to synthesize a single phylo-

genetic tree encompassing all life (OpenTreeOfLife et al. 2019). The OpenTree database Phylesystem (McTavish et al. 2015) is programmatically available free of cost to users through OpenTree's Application Programming Interface (API) services (OpenTreeOfLife et al. 2021). Additionally, software packages written with open source, free of cost programming languages commonly used in scientific research such as R and Python (Baker 2017) have been developed as wrappers that have increased the availability of OpenTree's API services, by making the OpenTree API functionalities more accessible to a wider user audience (Michonneau et al. 2016, Mctavish et al. 2021). Yet, the R and Python OpenTree API wrapper software packages have been mainly used by computer-literate individuals to seamlessly establish reproducible workflows to use and reuse expert phylogenetic knowledge for biological research (Sánchez-Reyes & O'Meara 2019, Sánchez-Reyes et al. 2021) and education (Nguyen et al. 2020, Jodie Wiggins and Phylotastic Team 2021, Matt Wilkins and Galactic Polymath 2021). In the 5 years since its release, the R package wrapper for OpenTree `rotl` has been cited by more than 190 biological research papers addressing various topics (Google Search February 02, 2022). Comparatively, other R packages for phylogenetics, such as `GGTREE` and `mixOmics`, which were released one year after `rotl` have been cited over a thousand times, which might indicate that `rotl` has not been adopted as widely.

While learners in the natural sciences have been engaging independently with R and Python programming language as they represent two of the most widely used programming languages in the sciences (Baker 2017), computer programming is not traditionally a core skill formally taught to biologists and naturalists (Sayres et al. 2018, Wright et al. 2019, Williams et al. 2019). However, the OpenTree project demonstrates that efforts to improve availability and reproducibility in phylogenetics ve increased required baseline computational knowledge and skills in the field. As computers continue to play a larger role in most scientific disciplines (Piccolo & Frampton 2016), baseline computational skill requirements are also increasing across all natural sciences. Thus, efforts to increase reproducibility rates in the natural sciences must consider the particularities of scientific workflows that rely on usage of programming languages, such as availability and accessibility of data and code (Peng 2011, Sandve et al. 2013, Powers & Hampton 2019).

We chose a phylogenetics workflow that relies on data and code from OpenTree to identify the specific barriers to accessibility and reproducibility it presents. Then, we design ways to overcome or diminish these barriers and apply them to a series of tutorials and vignettes developed for the OpenTree project. Finally, we generalize our finding to suggest a set of principles that can be used as guide to develop code and documentation materials with increased accessibility that can improve availability and contribute to reproducibility. Notably, these principles can be incorporated as learning goals into any syllabus for a course or workshop on best practices for scientific reproducibility.

Identifying hurdles to accessibility

Good primary documentation for code is thorough. It describes general usage of individual functions, the components and variables a function can take, and it should be accompanied with function usage examples on how to apply it (Karimzadeh & Hoffman 2018). As opposed to code, primary documentation is written in natural language (i.e., any known human language, e.g., English, Spanish, Chinese). Primary documentation is viewed as a key element for success of a piece of code (Karimzadeh & Hoffman 2018), which might be why it is also usually written using highly specialized computational jargon (i.e., computationally specific concepts, words, and phrases) as well as formal scientific language. While this might be important for formal acceptance of the code by the scientific and academic community, it often slows down or even obstructs examination, application, and adoption of code by the general audience (Ball 2017).

Vignettes and tutorials work as secondary pieces of documentation, that help to demonstrate additional cases of individual function usage, and showcase function associations that work for specific analysis workflows in more detail. As secondary documentation has become more common practice and is more flexible in its form and content, it constitutes an ideal canvas to develop, implement and test principles that can overcome current barriers to code accessibility. The tutorials are available at https://mctavishlab.github.io/R_OpenTree_tutorials/.

Addressing hurdles to accessibility: some principles

a. Literate programming: Demonstrate code usage with integrative examples

Pedagogical research shows that active learning practices are one the most effective ways to take on abstract subjects (Freeman et al. 2014). Programming computer languages are quite abstract and learning them can be greatly enhanced by applying an active learning strategy such as ~~a~~ linking its usage to a "real world" or "human" application (Felder & Brent 2009).

A story-like narrative that links pieces of code together and invites learners to try the code can ~~greatly support~~ learners to remember what they are doing and why they are doing it. This "literate programming" paradigm (Knuth 1984, Fritzson et al. 2002) makes code more approachable, as it integrates narratives with computer code in the same document, supporting learners in actively following, remembering and understanding the code usage (Piccolo & Frampton 2016). Documents developed with "literate programming" can be made more accessible by choosing narratives that are relatable to a more general audience.

We examined available primary documentation for the package `rotl`, and designed a narrative that required the usage of as many functions as possible. We demonstrate code applications that are commonly requested by OpenTree users, but that are not demonstrated in the R package primary documentation. By framing the function workflow using highly requested uses, the documentation acquires a narrative arc that is easier to follow and remember by users. This can also facilitate the application of code to other use cases in biology of interest for the learners.

b. Demonstrate errors and warnings thoroughly

A practice that has become more and more widespread in programming languages pedagogical practices is the use of typos and mistakes to normalize them for learners and show them how to solve them when they are outside the classroom (Shannon & Summet 2015). Yet, this is rarely done for written pedagogical materials. Primary documentation focuses on demonstrating usage function with examples that work seamlessly, without errors. We argue that the opposite is needed to support adoption of reproducible workflows and support

long term independence in learners (Gaspar & Langevin 2007). We demonstrate examples that do not work as expected and exemplify ways to address them (Figure 1).

We identify inputs that would give a wide range of warnings and errors, focusing on demonstrating these cases. This helps users to not be afraid of errors and warnings, but instead to use them to their advantage. We also identify effects of warnings and errors downstream of the workflow.

We identify ways to evaluate inputs to know if they will produce an error, and design alternatives on what to do when faced with an error or warning, and demonstrate these alternatives. One of the most essential skills in programming is interpreting and moving forward from errors. Many finely honed tutorials do not trigger errors, which precludes helping students to develop the tools to understand and address errors when they do encounter them, as they inevitably will. ~~On~~ our tutorial, we focus on explaining the meaning and downstream of warnings and errors, and showcase ways to detect them before they are triggered (i.e., before using an input that would elicit a warning or error). This has two pedagogical benefits: 1) it provides users/students with the means to troubleshoot their own warnings and errors, and 2) it allows users/students to understand with more depth what the function is doing.

c. Avoid jargon and expert language

Besides avoiding formal language, and incorporating elements of pop culture, such as picture character icons known as "emojis", to make the language more familiar to a broader target audience (see Figure 1), we made an effort to specifically complement the primary documentation by identifying computational concepts that were assumed or were not explained in depth. We vetted the tutorials with an audience on workshops as well as individual user. We choose examples that are charismatic for the audience. For example, when we presented the tutorial for a team specialized in Amphibians, we tailored the examples using frogs and their allies.

Now, let's extract a subtree for the genus *Canis*. It should be way smaller!

```R
subtree <- rotl::tol_subtree(resolved_names["Canis",]$ott_id)
```

```
Error: HTTP failure: 400
list(contesting_trees = list(`ot_278@tree1` = list(attachment_points =
list(list(children_from_taxon = list("node242"), parent = "node241"),
list(children_from_taxon = list("node244"), parent = "node243"), list
(children_from_taxon = list("node262"), parent = "node255"), list(chil
dren_from_taxon = list("node270"), parent = "node267"))), `ot_328@tree
1` = list(attachment_points = list(list(children_from_taxon = list("no
de519"), parent = "node518"), list(children_from_taxon = list("node52
3"), parent = "node522")))),
    mrca = "mrcaott47497ott110766")[/v3/tree_of_life/subtree] Error: n
ode_id was not found (broken taxon).
```

😱 😱 😱

What does this error mean??

A "broken" taxon error usually happens when phylogenetic information does not match taxonomic information.

Figure 1: Snapshot of a section of the tutorial website, where we demonstrate a common error.

d. Make it stable through time

We published the tutorials on a public, free license, free of cost, and free for use and reuse repository and persistent website (Sanchez-Reyes, Luna L and McTavish, Emily Jane and Holder, Mark T 2021*a*,*b*). The tutorial is available for the users to go back to ~~it~~ any time they need it, and to be passed on to other users (Figure 2).

Following the carpentries, we created a main version of the tutorial that is updated. Versions presented ~~on~~ workshops are a copy from the original repository, and represent a stable and temporal snapshot of the functions and workflows presented in the tutorial.

Conclusion

Ultimately, the long term improvement of reproducibility rates in science will depend on our ability to intentionally integrate the subject of reproducibility into the undergraduate curriculum, so college learners and future researchers have the basis to develop the fundamental skills needed to successfully create reproducible scientific workflows and materials.

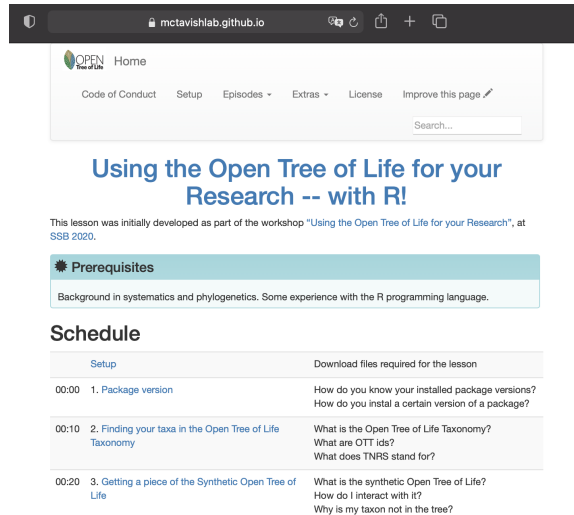Some universities have been incorporating the subject in their classes (see University of

Figure 2: Snapshot of the home to our tutorial website, showing part of the schedule.

Washington Libraries (2022), NIGMS Career Curriculum Development 2015 (2022)). The focus of this resources has been for students to develop skills to document their work. The principles identified and outlined here can be used to set learning goals and outcomes on new reproducibility syllabus.

We have received emails from senior researchers thanking us for this materials, and students have been able to engage using the packages with less hep from the PIs.

The principles to create tutorials described here facilitate adoption of software and analysis workflows among researchers at different academic levels, from undergrads to established researchers. It will also help closing the gap between students that had access to computational resources (and computational training) from an early age and students that did not. Late access to computational resources and training can occur due to lack of economic sources, often occurring in households from underrepresented communities and minorities. It can also be due to gender-biased parental and community pressures, in which male individuals are more often encouraged to perform activities related to computers, while female individuals are discouraged. These principles can be used to aide not only in improving reproducibility practices, but also software adoption in the natural sciences.

Making accessible reproducible workflows has several advantages: save explanation/training time when analyses are run again by students and collaborators. save research time for

yourself when analyses are run again with more data, a different dataset, a different organism or biological model. scientific efforts can build off of each other

SUPPLEMENTARY MATERIAL

Title: Website and GitHub repository containing the complete teaching materials developed and demonstrated here.

GitHub repository link: `https://github.com/McTavishLab/R_OpenTree_tutorials`

Website link: `https://mctavishlab.github.io/R_OpenTree_tutorials`

References

Arnold, B., Bowler, L., Gibson, S., Herterich, P., Higman, R., Krystalli, A., Morley, A., O'Reilly, M., Whitaker, K. et al. (2019), 'The turing way: a handbook for reproducible data science (version v1.0.1)', *Zenodo* .

Baker, M. (2016), 'Is there a reproducibility crisis?', *Nature* **533**(26), 353–66.

Baker, M. (2017), 'Scientific computing: Code alert', *Nature* **541**(7638), 563–565.

Ball, P. (2017), 'It's not just you: science papers are getting harder to read', *Nature* **30**.

Cambridge Dictionary, (2022), 'Availability', *Cambridge University Press* .
  URL: *dictionary.cambridge.org/us/dictionary/english/availability*

Curating for Reproducibility Consortium (2021), 'Defining "reproducibility"'.
  URL: *https://cure.web.unc.edu/defining-reproducibility/*

Dobzhansky, T. (1973), 'Nothing in biology makes sense except in the light of evolution', *The American Biology Teacher* **35**(3), 125–129.

Fecher, B. & Friesike, S. (2014), *Open Science: One Term, Five Schools of Thought*, Springer International Publishing, pp. 17–47.

Felder, R. M. & Brent, R. (2009), 'Active learning: An introduction', *ASQ higher education brief* **2**(4), 1–5.

Freeman, S., Eddy, S. L., McDonough, M., Smith, M. K., Okoroafor, N., Jordt, H. & Wenderoth, M. P. (2014), 'Active learning increases student performance in science, engineering, and mathematics', *Proceedings of the national academy of sciences* **111**(23), 8410–8415.

Fritzson, P., Gunnarsson, J. & Jirstrand, M. (2002), Mathmodelica - an extensible modeling and simulation environment with integrated graphics and literate programming, *in* '2nd International Modelica Conference, March 18-19, Munich, Germany'.

Gaspar, A. & Langevin, S. (2007), Restoring "coding with intention" in introductory programming courses, *in* 'Proceedings of the 8th ACM SIGITE conference on Information technology education', pp. 91–98.

Google Search (February 02, 2022), 'References citing the paper rotl: an R package to interact with the Open Tree of Life data'.
URL: *https://scholar.google.com/scholar?cites=15203967220597193208*

Ioannidis, J. P. (2005), 'Why most published research findings are false', *PLoS medicine* **2**(8), e124.

Jodie Wiggins and Phylotastic Team (2021), 'Scientific data in your classroom'.
URL: *https://jwiggi18.github.io/phyloEd/*

Karimzadeh, M. & Hoffman, M. M. (2018), 'Top considerations for creating bioinformatics software documentation', *Briefings in Bioinformatics* **19**(4), 693–699.

King, G. (1995), 'Replication, replication', *PS: Political Science & Politics* **28**(3), 444–452.

Knuth, D. E. (1984), 'Literate programming', *The computer journal* **27**(2), 97–111.

Matt Wilkins and Galactic Polymath (2021), 'support K-16 education and scicomm'.
URL: *https://github.com/galacticpolymath/galacticEdTools*

McTavish, E. J., Hinchliff, C. E., Allman, J. F., Brown, J. W., Cranston, K. A., Holder, M. T., Rees, J. A. & Smith, S. A. (2015), 'Phylesystem: a git-based data store for community-curated phylogenetic estimates', *Bioinformatics* **31**(17), 2794–2800.

Mctavish, E. J., Sánchez-Reyes, L. L. & Holder, M. T. (2021), 'Opentree: A python package for accessing and analyzing data from the Open Tree of Life', *Systematic Biology* .
URL: *https://doi.org/10.1093/sysbio/syab033*

Michonneau, F., Brown, J. W. & Winter, D. J. (2016), 'rotl: an R package to interact with the Open Tree of Life data', *Methods in Ecology and Evolution* **7**(12), 1476–1481.

Nguyen, V. D., Nguyen, T. H., Tayeen, A. S. M., Laughinghouse IV, H. D., Sánchez-Reyes, L. L., Wiggins, J., Pontelli, E., Mozzherin, D., O'Meara, B. & Stoltzfus, A. (2020), 'Phylotastic: improving access to tree-of-life knowledge with flexible, on-the-fly delivery of trees', *Evolutionary Bioinformatics* **16**, 1176934319899384.

NIGMS Career Curriculum Development 2015 (2022), 'Rigor & Reproducibility, National Institute of General Medical Sciences'.
URL: *https://www.nigms.nih.gov/training/instpredoc/Pages/admin-supplements-prev.aspx*

OpenTreeOfLife, Redelings, B., Cranston, K. A., Allman, J., Holder, M. T. & McTavish, E. J. (2021), 'Open Tree of Life APIs v. 3.0'.
URL: *https://github.com/OpenTreeOfLife/germinator/wiki/Open-Tree-of-Life-Web-APIs*

OpenTreeOfLife, Redelings, B., Reyes, L. L. S., Cranston, K. A., Allman, J., Holder, M. T. & McTavish, E. J. (2019), 'Open tree of life synthetic tree'.
URL: *https://doi.org/10.5281/zenodo.3937742*

Peng, R. (2015), 'The reproducibility crisis in science: A statistical counterattack', *Significance* **12**(3), 30–32.

Peng, R. D. (2011), 'Reproducible research in computational science', *Science* **334**(6060), 1226–1227.

Piccolo, S. R. & Frampton, M. B. (2016), 'Tools and techniques for computational reproducibility', *Gigascience* **5**(1), s13742–016.

Powers, S. M. & Hampton, S. E. (2019), 'Open science, reproducibility, and transparency in ecology', *Ecological Applications* **29**(1), e01822.

Prinz, F., Schlange, T. & Asadullah, K. (2011), 'Believe it or not: how much can we rely on published data on potential drug targets?', *Nature reviews Drug discovery* **10**(9), 712–712.

Sánchez-Reyes, L. L., Kandziora, M. & McTavish, E. J. (2021), 'Physcraper: a python package for continually updated phylogenetic trees using the open tree of life', *BMC bioinformatics* **22**(1), 1–13.

Sánchez-Reyes, L. L. & O'Meara, B. C. (2019), 'Datelife: Leveraging databases and analytical tools to reveal the dated tree of life', *bioRxiv* p. 782094.

Sanchez-Reyes, Luna L and McTavish, Emily Jane and Holder, Mark T (2021*a*), 'McTavishLab/R_OpenTree_tutorials v0.9.1: Using the Open Tree of Life for your Research, with R'.
URL: *https://github.com/McTavishLab/R_OpenTree_tutorials*

Sanchez-Reyes, Luna L and McTavish, Emily Jane and Holder, Mark T (2021*b*), 'Using the Open Tree of Life for your Research, with R'.
URL: *https://mctavishlab.github.io/R_OpenTree_tutorials/*

Sandve, G. K., Nekrutenko, A., Taylor, J. & Hovig, E. (2013), 'Ten simple rules for reproducible computational research', *PLoS computational biology* **9**(10), e1003285.

Sayres, M. A. W., Hauser, C., Sierk, M., Robic, S., Rosenwald, A. G., Smith, T. M., Triplett, E. W., Williams, J. J., Dinsdale, E., Morgan, W. R. et al. (2018), 'Bioinformatics core competencies for undergraduate life sciences education', *PloS one* **13**(6), e0196878.

Shannon, A. & Summet, V. (2015), 'Live coding in introductory computer science courses', *Journal of Computing Sciences in Colleges* **31**(2), 158–164.

University of Washington Libraries (2022), 'Teaching Reproducibility'.
URL: *https://guides.lib.uw.edu/research/reproducibility/teaching*

Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E. et al. (2016), 'The fair guiding principles for scientific data management and stewardship', *Scientific data* **3**(1), 1–9.

Williams, J. J., Drew, J. C., Galindo-Gonzalez, S., Robic, S., Dinsdale, E., Morgan, W. R., Triplett, E. W., Burnette III, J. M., Donovan, S. S., Fowlks, E. R. et al. (2019), 'Barriers to integration of bioinformatics into undergraduate life sciences education: A national study of us life sciences faculty uncover significant barriers to integrating bioinformatics into undergraduate instruction', *PLoS One* **14**(11), e0224288.

Wright, A. M., Schwartz, R. S., Oaks, J. R., Newman, C. E. & Flanagan, S. P. (2019), 'The why, when, and how of computing in biology classrooms', *F1000Research* **8**.