Teaching analysis reproducibility with accessibility: Enriching secondary documentation to support skill level diversity of users.

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 11, 2022

Abstract

Research reproducibility is essential for scientific development. Yet, rates of reproducibility are low, especially in the natural sciences. As increasingly more research is relying on computing tools and software, efforts for improving reproducibility rates have focused on digital availability of research workflows as computer code, as well as raw and processed data in computer readable form. However, making research products digitally available does not necessarily ensures that they can be easily reached by the majority of a target user base, let alone the general population, making them effectively inaccessible, counteracting availability, and hindering reproducibility. Thus, accessibility of research products is a key aspect for the long term improvement of reproducibility rates. Using an example within evolutionary biology, we identify inclusivity barriers to accessibility, research products in the natural sciences: use of highly specialized computing jargon and programming techniques; high cognitive load, unspecified, unclear or lengthy goals; content-focused descriptions instead of user-focused; inflexible learning environment; unapproachable (cold or intimidating) language; and little to no diversity of representation of information.

Then, we propose a set of principles to improve accessibility of code and data, and illustrate their application in a set of teaching materials for evolutionary biology that we put to work in the field. Finally, we elaborate on the general application of these principles for documenting scientific workflows, to provide present learners and future researchers with tools for successful scientific reproducibility.

1

Introduction

Research reproducibility –the extent to which consistent results are obtained when a scientific experiment or research workflow is repeated (Curating for Reproducibility Consortium 2017)– 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 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 a property denoting that a research product can be reached (acquired, copied, analyzed, processed and/or reused) at no financial, legal or technical cost (Arnold et al. 2019), and without geographic, demographic, social 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 recognize that the term accessibility is used in relation to various aspects of the human experience. In relation to digital resources, these aspects have been broadly categorized in four axis: perception, operability, understanding and robustness (Chisholm et al. 2001, Caldwell et al. 2008) Some forms of accessibility do not support automatization nor scalability. For example, Graphical User Interfaces (GUIs) are accessible, but not scalable and rarely automatizable.

we focus on X types of accessibility

We identify factors that have particularly affected accessibility in the natural sciences, and ways the scientific community can address them to create reproducible research workflows. These principles can be generalized and integrated into the undergraduate and graduate school STEM curriculum, either for courses specialized in reproducibility or within other subject areas, as a necessary component of successful and impactful science.

Box 1. **The role of accessibility in improving reproducibility.** Availability and accessibility are conceptually accepted as synonyms for "the fact of being able to be reached" (Cambridge Dictionary 2022b). However, accessibility's definition goes further and describes it as the "fact of being able to be reached **easily**" (Cambridge Dictionary 2022a). A secondary definition of accessibility elaborates more on this: "the quality of being **easy to understand**".

What we realize from these definitions is that, in practice, something might be available but not necessarily accessible, hindering the realization of availability. To illustrate the practical difference between availability and accessibility we will use an allegory that goes like this:

** Change marshmallow story to cake

** Explain different types of barriers to accessibility within the example, then within science

** Explain why we focus on psychological/social/pedagogical barriers

** Gauging accessibility of a resource is tricky because it depends on an individual experience of what is in effect **easy** for someone. **Easy** is a subjective quality that fully depends on individual perceptions resulting from individual contexts. As a society we are moving towards validating and welcoming all human experiences. To improve accessibility levels in the natural sciences, it is important to account for the factors that are creating different experiences of what is **easy**.

** Now, explain different types of barriers to accessibility in science (Jessica: Maybe say that lack of accessibility can be due to a number of factors (physical, resources, economic), but one critical factor is the use of impenetrable expert language.)

An example from phylogenetics

To elaborate on our thesis, we choose an example from the research field of phylogenetics, a discipline within evolutionary biology. Phylogenetics focuses on investigating the history of shared ancestry of living organisms from biological data, and represents this evolutionary history in a diagram known as a phylogeny (Figure 1). Phylogenies provide the basis to study and understand all biological processes in an evolutionary context (Dobzhansky

1973). Hence, it appears that improving reproducibility rates in phylogenetics has the potential to positively impact research across the natural sciences.

***What other platforms provide resources for reproducibility in phylogenetics???

The Open Tree of Life project (OpenTree) has developed a computational platform that provides digital availability of phylogenetic results from vetted, state-of-the-art research in the field (Open Tree Of Life et al. 2019). OpenTree's database, the Phylesystem (McTavish et al. 2015), provides usability of phylogenetic data by standardizing organism names across biological databases, and supporting digital data obtention in a computer readable format.

While this is accessible through their Graphical User Interface, It does not support reproducibility

Accessibility does not imply reproducibility

In order for something to be highly reproducible, it has to be largely accessible.

is programmatically available free of cost to users through OpenTree's Application Programming Interface (API) services (Open Tree Of Life et al. 2016). 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, Wiggins & Phylotastic Team 2019, Wilkins & 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 Scholar Search - Feb 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 a delay `rotl` adoption. Of course one potential explanation is that OpenTree services do not have as wide an application as services provided by the afore mentioned R packages. However, the expressed enthusiasm (interest to use/inability to use the resource) by the

target audience and larger community might point to a different issue.

While learners in the natural sciences have been engaging independently with R and Python programming languages 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). Moreover, the main usage of the OpenTree API services hints that tools intended to improve availability and reproducibility come at a cost for the potential users: an increased required baseline computational knowledge and skillset in the field. As computers continue to play a larger role in most scientific disciplines (Piccolo & Frampton 2016), higher baseline computational skills are required across all natural sciences. Thus, efforts to increase reproducibility rates in the natural sciences must consider the specifics 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 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. The tutorials are available at https://mctavishlab.github.io/R_OpenTree_tutorials/.

Finally, we generalize our findings to suggest a set of principles that can be used as a guide to develop code and documentation materials with increased accessibility, to 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
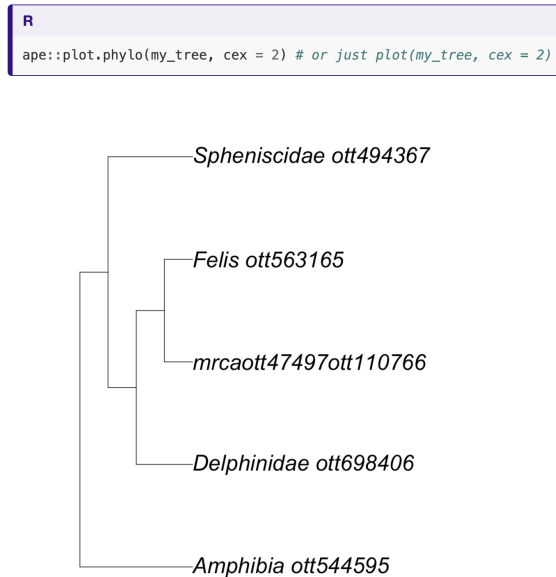
```R
ape::plot.phylo(my_tree, cex = 2) # or just plot(my_tree, cex = 2)
```

```
┌──────── Spheniscidae ott494367

         ┌──── Felis ott563165
    ┌────┤
    │    └──── mrcaott47497ott110766
────┤
    │    └──── Delphinidae ott698406

    └──────── Amphibia ott544595
```

Figure 1: A phylogenetic tree from our tutorial website, extracted using OpenTree of Life tools and the rotl R package.

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.

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 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 lead 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 the code usage, supporting memory and understanding (Piccolo & Frampton 2016).

We propose that documents developed with "literate programming" can be made more accessible by choosing narratives that are relatable to a more general audience. An easy way to do this in biology is choosing as model organism a taxon that a research group is more interested in studying. For the non-specialized user base, a highly charismatic group such as dinosaurs will do the trick.

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 primary documentation of the R package. 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 users.

b. Demonstrate errors and warnings thoroughly

A practice that has become more and more widespread in programming-language 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 on 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 2).

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. In 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 2), 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 through feedback from workshop participants as well as individual users. We choose examples that are charismatic for the audience. For example, when we presented the tutorial for a team specialized in Amphibians, we tailored

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

the examples using frogs and their allies.

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 (Sánchez Reyes, Luna L and McTavish, Emily Jane and Holder, Mark T 2021). The tutorial is available for the users to go back to any time they need it, and to be passed on to other users (Figure 3).

We created a main version of the tutorial that is stable. Any updates to the tutorial are published as new versions, or tutorials for new workshops (Wilson 2006, 2022) Versions presented at workshops are a copy from the original repository. They represent a temporally stable snapshot of functions and workflows presented during a workshop.

Conclusion

Response form the community has been invaluable in gauging success of the tutorials. Senior researchers often comment on the usefulness of the tutorials for their research, as well as supporting students in using the R packages with less help from them as PIs.
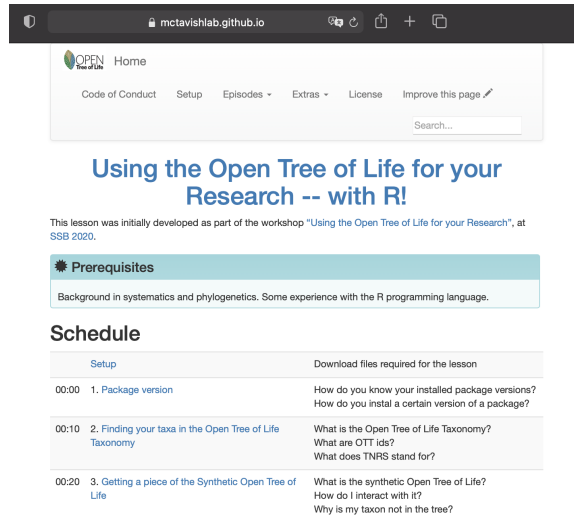
10

Figure 3: Snapshot of the home to our tutorial website, showing part of the schedule.

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.

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 Washington Libraries - (2022), NIGMS Career Curriculum Development - (2015)). The focus of these 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 syllabi.

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 close 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 resources, 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 improve not only reproducibility practices, but also software adoption in the natural sciences.

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**.

Caldwell, B., Cooper, M., Reid, L. G., Vanderheiden, G., Chisholm, W., Slatin, J. & White, J. (2008), 'Web content accessibility guidelines (wcag) 2.0', *WWW Consortium (W3C)* **290**, 1–34.

Cambridge Dictionary (2022*a*), 'Accessibility', *Cambridge University Press* .
    URL: *https://dictionary.cambridge.org/us/dictionary/english/accessibility*

Cambridge Dictionary (2022b), 'Availability and Accessibility', *Cambridge University Press* .
  URL: *https://dictionary.cambridge.org/es/diccionario/ingles/accessibility?topic=available-and-accessible*

Chisholm, W., Vanderheiden, G. & Jacobs, I. (2001), 'Web content accessibility guidelines 1.0', *Interactions* **8**(4), 35–54.

Curating for Reproducibility Consortium (2017), '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 Scholar Search - (Feb 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.

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.

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

Open Tree Of Life, Redelings, B., Cranston, K. A., Allman, J., Holder, M. T. & McTavish, E. J. (2016), 'Open Tree of Life APIs v3.0', *Open Tree of Life Project* (Online Resources).
URL: *https://github.com/OpenTreeOfLife/germinator/wiki/Open-Tree-of-Life-Web-APIs*

Open Tree Of Life, Redelings, B., Sánchez Reyes, L. L., Cranston, K. A., Allman, J., Holder, M. T. & McTavish, E. J. (2019), 'Open tree of life synthetic tree v12.3', *Zenodo* .

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.

Sánchez Reyes, Luna L and McTavish, Emily Jane and Holder, Mark T (2021), 'Using the Open Tree of Life for your Research, with R v0.9.1', *Open Tree of Life* (Online Resources).
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*

Wiggins, J. & Phylotastic Team (2019), 'Phylotastic Ed: Scientific Data in Your Classroom', *Phylotastic Project* (Online Resources).
  URL: *https://jwiggi18.github.io/phyloEd/*

Wilkins, M. & Galactic Polymath (2021), 'galacticEdTools: Tools for Inspiring K-16 Students to Become Interdisciplinary Experts and Agents of Change for a Better Tomorrow', *R Package in Development* (GitHub repository).
  URL: *https://github.com/galacticpolymath/galacticEdTools*

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.

Wilson, G. (2006), 'Software Carpentry: Getting Scientists to Write Better Code by Making Them More Productive', *Computing in Science & Engineering* .

Wilson, G. (2022), 'The Carpentries', *Web Site* .
  URL: *http://software-carpentry.org*

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**.