Not just for programmers: How GitHub can accelerate collaborative and reproducible research in ecology and evolution

This manuscript ([permalink](#)) was automatically generated from [SORTEE-Github-Hackathon/manuscript@8d4190c](#) on November 17, 2022.

Authors

- **Robert Crystal-Ornelas**
  [0000-0002-6339-1139](#) · [robcrystalornelas](#) · [rob_c_ornelas](#)
  Earth and Environmental Sciences Area, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

- **Brandon P.M. Edwards**
  [0000-0003-0865-3076](#) · [BrandonEdwards](#)
  Department of Biology, Carleton University, Ottawa, ON K1S 5B6, Canada

- **Katherine Hébert**
  [0000-0001-7866-6775](#) · [katherinehebert](#) · [hebert_kat](#)
  Département de biologie, Université de Sherbrooke, Québec, Canada

- **Emma J. Hudgins**
  [0000-0002-8402-5111](#) · [emmajhudgins](#)
  Department of Biology, Carleton University, Ottawa, ON K1S 5B6, Canada

- **Luna L. Sánchez Reyes**
  [0000-0001-7668-2528](#) · [LunaSare](#) · [LunaSare](#)
  School of Natural Sciences, University of California, Merced, USA

- **Eric R. Scott**
  [0000-0002-7430-7879](#) · [Aariq](#)
  Department of Wildlife Ecology and Conservation, University of Florida, Gainesville, FL, USA

- **Matthew J. Grainger**
  [0000-0001-8426-6495](#) · [DrMattG](#) · [Ed_pheasant](#)
  Terrestrial Biodiversity, Norwegian Institute for Nature Research - NINA, Postbox 5685 Torgarden, 7485 Trondheim, Norway

- **Vivienne Foroughirad**
  [0000-0002-8656-7440](#) · [vjf2](#) · [vforoughirad](#)
  Department of Biology, Georgetown University, Washington, DC, USA

- **Allison D. Binley**
  [0000-0001-8790-9935](#) · [adbinley](#) · [AllisonBinley](#)
  Department of Biology, Carleton University, Ottawa, ON K1S 5B6, Canada

- **Cole B. Brookson**
  [0000-0003-1237-4096](#) · [colebrookson](#)
  Department of Biological Sciences, University of Alberta, Edmonton, AB, Canada

- **Kaitlyn M. Gaynor**
  0000-0002-5747-0543 · kaitlyngaynor · kaitlyngaynor
  Departments of Zoology and Botany, University of British Columbia, Vancouver, BC, Canada; National Center for Ecological Analysis and Synthesis, Santa Barbara, CA 93101, USA

- **Saeed Shafiei Sabet**
  0000-0001-5919-2527 · shafieisabets · SaeedSHSABET
  Fisheries Department, Faculty of Natural Resources, University of Guilan, Sowmeh Sara, Iran

- **Ali Güncan**
  0000-0003-1765-648X · Aguncan · aliguncan
  Department of Plant Protection, Faculty of Agriculture, Ordu University, 52200, Ordu, Turkey

- **Friederike Hillemann**
  0000-0002-8992-0676 · fhillemann
  Department of Human Behavior, Ecology and Culture, Max Planck Institute for Evolutionary Anthropology, Leipzig, Germany

- **Helen Weierbach**
  0000-0001-6348-9120 · helenweierbach · HWeierbach
  Earth and Environmental Sciences Area, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

- **Dylan G. E. Gomes**
  0000-0002-2642-3728 · dylangomes
  (Current) National Academy of Sciences NRC Research Associateship Program, Northwest Fisheries Science Center, National Marine Fisheries Service, National Oceanic and Atmospheric Administration, Seattle, WA, USA 98112; (Former) Cooperative Institute for Marine Resources Studies, Hatfield Marine Science Center, Oregon State University, Newport, OR, USA 97365

- **Pedro Henrique Pereira Braga**
  0000-0002-1308-1562 · pedrohbraga · pedrohp_braga
  Department of Biology, Concordia University, 7141 Sherbrooke Street West, Montreal, QC H4B 1R6, Canada

Abstract

Researchers in ecology and evolutionary biology are increasingly dependent on computational code to conduct research, and the use of efficient methods to share, reproduce, and collaborate on code as well as any research-related documentation has become fundamental. GitHub is an online, cloud-based service that can help researchers track, organize, discuss, share, and collaborate on software and other materials related to research production, including data, code for analyses, and protocols.

Despite these benefits, the use of GitHub in ecology and evolution is not widespread. To help researchers in ecology and evolution adopt useful features from GitHub in their own workflows, we review thirteen practical ways to use the platform. We outline features ranging from low to high technical difficulty: storing code, managing projects, coding collaboratively, conducting peer review, and writing a manuscript. Given that members of a research team may have different technical skills and responsibilities, we describe how the optimal use of GitHub features may vary among members of a research collaboration. As more ecologists and evolutionary biologists establish their workflows using GitHub, the field can continue to push the boundaries of collaborative, transparent, and open research.

Introduction

Most scientists, including ecologists and evolutionary biologists, are increasingly dependent on computational tools in their research (Hannay et al., 2009). Researchers write and use software packages or data analysis code (hereafter, code) to perform scientific tasks ranging from data management, data analysis, and study replication, to the application and the development of tools for hypothesis testing. Maintaining code for scientific collaboration requires an efficient and well-documented work-flow (J. M. Perkel, 2020). To facilitate this process, scientists have been adopting tools from information and systems technology, such as cloud-based services for documentation and version control (*e.g.*, from the Google Suite, the Microsoft Suite, and GitHub (J. Perkel, 2016)). However, most researchers lack exposure to adequate code practices and thus dedicate valuable time and effort to self-teaching research-facilitating tools. Thus, researchers may not adhere to standards of code quality and maintenance (Hannay et al., 2009; Prabhu et al., 2011; Wilson et al., 2014). Here, we review and discuss one of the most used web-based platforms for computational version control and collaboration, GitHub, and provide researchers in ecology and evolutionary biology (EEB) with practical workflows to facilitate and improve their code and its management.

With over 83 million registered users as of 2022, GitHub (https://github.com) is the most widely-used web platform for collaborating on computer code (*Build Software Better, Together*, n.d.). GitHub provides a simple but powerful web interface that allows users to participate in projects by contributing, modifying and discussing existing code, reporting bugs, discovering code and data, and publishing new code. Through version control, users have a detailed, chronological record of the files and directories stored in their repositories (Bryan, 2018) (see Box 1). This workflow provides a strong and clear advantage over sending files back-and-forth (*e.g.*, via email), a process that can become challenging and time-consuming in long-term and collaborative projects (Ram, 2013). Through its combination of version control and collaborative features, GitHub facilitates open-source code alongside collaborative development (Perez-Riverol et al., 2016).

Git is the version control system that enables all the collaborative tools available on GitHub. Although the understanding of basic concepts of Git (such as commit, push, pull, checkout; see Box 1) is necessary, the GitHub web-based platform and its integrated development environments (such as the GitHub Desktop) allow users to perform most repository and data management operations without opening Git command-line sessions.

The expansive GitHub user-community and numerous GitHub resources have boosted its popularity [Perez-Riverol et al. (2016); Bryan (2018); https://happygitwithr.com; https://ourcodingclub.github.io]. Nevertheless, although multiple articles have encouraged researchers in EEB to adopt GitHub as part of their research process (Lowndes et al., 2017; J. Perkel, 2016), its use is still not widespread. First-time users without formal training in information technology may face steep learning curves because GitHub and its features have been centered on collaboration for code development in information systems (Leibzon, 2016). Moreover, domain-specific resources providing tractable examples and practical guidance for researchers in EEB on GitHub are scarce [but see Kim et al. (2022); https://ourcodingclub.github.io; https://www.openscapes.org]. Widespread adoption of GitHub for collaborating on research tasks can ultimately enable EEB researchers to spend less time on creating novel processes for collaboration and more time on their research (Briney et al., 2020). More importantly, expanding the availability of data and code management standards – of which GitHub is one increasingly important component – makes research more reproducible and collaborative (Alston & Rick, 2021; Gomes et al., 2022).

This paper is the result of an academic hackathon held during the 2021 conference for the Society for Open, Reliable, and Transparent Ecology and Evolutionary Biology (SORTEE, https://www.sortee.org). We convened around thirty researchers in EEB with varying levels of familiarity with the usage of

GitHub in research projects to showcase and discuss how existing features can contribute to documentation and collaboration in EEB research. During the hackathon, we identified the need for formal discussions on how EEB researchers can benefit from GitHub and its features. Here, we outline thirteen practical ways that EEB researchers can use GitHub features for more collaborative, transparent, and reproducible science. We also provide critical perspectives on features that could be improved and catered towards research development.

Box 1

Glossary
repository: A repository (commonly shortened to "repo") is a collection of files (*e.g.*, a directory) tracked by Git. Repositories are managed by an owner and can be made either "public", to be visible to all GitHub users, or "private", to selected owner-specified users. Repositories can be either "local" and saved on an individual's computer or "remote" and stored on the cloud via GitHub's web platform.
fork: A fork is a copy of a **repository** hosted on GitHub. If a repository is public, then anyone can make a fork. Even if they do not have access to **push** to the original repository, they can make a fork and edit it independently. Forks are linked to the original GitHub repository and "upstream" changes (*i.e.*, those in the original repository) can be **merged** to keep the fork up-to-date with the original project. Changes made in the fork can be integrated into the original project via **pull requests**.
clone: Cloning a **repository** is a way of making a local copy (*i.e.*, on your computer) of a GitHub **repository**. If you have access to **push** to a **repository**, this can be a first step to contributing to a project.
branch: Git workflow timelines or repositories are analogous to trees, with a main working project and diverging branches that are pointers to changes during the development process. A git branch is an alternative line of development for a project (repository). Branches allow users to add new features or modifications to the project without affecting the main part of the project. Development branches can be created at any point in time and work on each branch can continue independently. Branching is useful for testing out new ideas (both code and text) which may or may not eventually get integrated into the main branch of the project. Branches can also be used to isolate contributions of multiple contributors. Each person working on their own branch eliminates problems that may arise if conflicting edits are **pushed** to the same remote branch. Changes in a development branch can be **merged** into the main branch via **pull requests**. Branches can only be made by those who are given access to the project **repository**.
commit: Commits are snapshots of the development of a project. In Git, versions of files and directories are uniquely identified as "commits", allowing one to identify and track modifications line-by-line. Commits can include changes in multiple files and must include a brief commit message describing the changes made. A typical workflow is to make some related changes in files, add a commit message (*e.g.*, "Generate and include results figure"), and after several commits **push** those commits to the remote (*i.e.*, cloud-based) GitHub **repository**.
push and pull: When **commits** are made in a project locally, they must be synced with the remote GitHub repository by **pushing** them. Changes on a GitHub repository can then be **pulled** to keep your local version of the project up-to-date with the remote.
pull request: A pull request is a request for changes made on an individual's **branch** in the repository or in a user's **fork** to be merged to the repository. Pull requests contain a description of the changes alongside all code required for testing and review by other users prior to being merged into the repository.
merge: Combining **commits** from two different branches together into one **branch**.
release: At any point a release can be made on GitHub to mark a significant milestone in the progression of a **repository**. While this GitHub feature is designed with releases of new versions of code in mind (*e.g.*, v1.0.0), it can also be used to create a snapshot of a repository at significant stages like pre-print, submission, revision, and acceptance of an associated manuscript.
community: A forum where GitHub users can ask for advice, offer solutions to questions, and share ideas (https://github.community/).

Thirteen practical ways GitHub can accelerate research in ecology and evolution

Storing and sharing research compendia

An EEB research compendium includes all computational materials related to research production, including data, code for analyses and protocols. Having copies of these files safely stored is essential to protect against accidental modifications or deletions. Many researchers begin using GitHub to store (or backup) their research compendium (Marwick et al., 2018) to a centralized, readily-available remote server (see Box 1). A centralized research compendium stored in a version-controlled repository has the advantages of facilitating collaboration, integrating data and code archiving services, allowing file versions to be accessed and restored, and contributes to open science (see sections below).

GitHub limits committed file sizes to 100 Mb (megabytes) (*About Large Files on GitHub*, n.d.), which can make it challenging for centralizing research compendia containing larger file sizes. Users may still version large files using Git Large File Storage (https://git-lfs.github.com) text pointers, but may have to rely on external file storage alternatives (such as local or cloud-hosting).

Virtual laboratory notebooks

Laboratory notebooks help researchers track their research notes, methods, policies and protocols (Kanza et al., 2017). Virtual laboratory notebooks can be stored in GitHub repositories and provide the benefits of simultaneous, centralized and selective access, and allows for the easy update of policies and experiment protocols (Schnell, 2015). Researchers have been increasingly using GitHub to maintain versions and share digital laboratory notebooks across many fields (J. Perkel, 2016), including applied ecology (*e.g.*, https://scheuerell-lab.github.io/lab-book), biogeography and global change biology (*e.g.*, https://github.com/HuckleyLab/how_we_work), and microbial ecology (*e.g.*, https://github.com/CarBBAS/uqam-guide).

At least for aspects of a research project that involve writing code, a GitHub repository is a form of a laboratory notebook; when changes are made to files in a version-controlled repository, the author of those changes makes a commit (Box 1) accompanied by a description of changes. Later, the entire history of commits and their commit messages are viewable and can be audited similar to a physical laboratory notebook (Ram, 2013). GitHub issues (see Project management section below) can be used to prioritize laboratory objectives and goals, as well as track any progress updates.

Project management

Modern research in ecology and evolution is highly collaborative, bringing together multidisciplinary teams from various institutions (Goring et al., 2014). On GitHub, collaborators can share feedback, brainstorm ideas, and troubleshoot problems (Figure 1). Project management can happen via three GitHub repository features: "Issues", "Discussion" and "Projects". Github Issues allow for discrete tasks and sub-tasks to be identified, assigned to team members, and categorized with custom labels. Github Discussions serve as a message board for conversation. Finally, GitHub Projects integrate issues and pull requests on automated spreadsheets and project boards, providing users with real-time tracking of project priorities and status (*About Projects*, n.d.). Scripts, commit messages, and pull requests can be linked directly to issues, discussions, and projects providing a clear record of project workflow. For instance, one can create a discussion on a project repository to decide which method to apply for biodiversity data analysis. Then, an issue can be created to establish steps and responsibilities including data formatting, statistical analyses, figure generation, and issue resolution (*e.g.*, see issues and discussions in the sPlotOpen manuscript repository, **doi:10.1111/geb.13346?**); *Issues · Fmsabatini/sPlotOpen_Manuscript* (n.d.)=is%3Aissue+is%3Aclosed]. Using GitHub for all project-related conversation and planning, rather than email or messaging tools, makes it easier to keep track of progress throughout the lifespan of a project. Unlike emails and messages which can get

lost as more new tasks arise, GitHub issues are intentionally closed by repository administrators hiding the issue from view (closed issues remain accessible but not immediately visible). Project management in GitHub can also be integrated with third-party applications, such as Zenhub (https://www.zenhub.com) or Slack (https://slack.github.com). Zenhub allows for the enhanced visualization and organization of repositories and their issues, while the GitHub for Slack integration allows notifications from GitHub events to be sent directly to users or group channels (see Slack, n.d.).

Educational materials

GitHub supports a broad set of mechanisms for hosting educational materials. The entire process of running a course, workshop, or lecture, can all be done openly on GitHub including material development, web hosting, and delivery, and even submission and grading of assignments. While there are other purpose-built platforms for this, GitHub provides a free, open-source alternative.

Making presentations, syllabi and other course materials can be done through most major high-level programming languages such as R, with `RMarkdown` (Xie et al,, 2019), Python, with `python-ppt` (https://python-pptx.readthedocs.io), and Julia, with `Remark.jl` (https://juliapackages.com/p/remark), and be version-controlled and stored in GitHub. Once content is made, hosting a course website can be done through GitHub Pages (https://pages.github.com; *e.g.*, https://github.com/topics/course-website). This way, course content can be available to enrolled students, as well as a global pool of learners and teachers interested in the material. Content can then be delivered via the course website, and/or a GitHub Organization with, for example, template repositories for assignments. Student submissions are perhaps the most challenging component, but the new GitHub Classroom tool (https://classroom.github.com) allows instructors to host private assignments to be submitted collaboratively or individually as code or PDF files, and even build autograding tests. Although time-consuming to establish, using these features can integrate learning version control and GitHub with the learning course content, and thus boost students' feelings of self-efficacy and confidence (Trujillo & Tanner, 2014).

Hosting a website

Personal or laboratory websites can improve the sharing of research findings, build online presence, and increase coordination of research efforts (Smaglik, 2007). Despite researchers in ecology and evolution generally lacking experience in building or hosting webpages, many tools have been developed to help this process. Static websites can now be easily built using independent software and languages, such as Jekyll (https://jekyllrb.com), Hugo (https://gohugo.io), Quarto (https://quarto.org), and wowchemy (https://wowchemy.com), or with the help of dependencies in the programming languages scientists commonly use, such as the `distill` (https://github.com/rstudio/distill) and `blogdown` (https://bookdown.org/yihui/blogdown) R packages. The resulting files from static websites can then be hosted in repositories, from which one can activate GitHub's Pages (https://pages.github.com) feature, allowing for the direct live hosting of HyperText Markup Language (HTML), Cascading Style Sheets (CSS) and JavaScript files within a free github.io domain.

Website templates are readily available on GitHub (https://github.com/topics/website-template) or in user-specific repositories, which can be forked (Box 1) and customized (Dawson, 2016). Aside from free hosting services, GitHub Pages also allows websites to be autogenerated and instantly updated following content modifications (Bryan, 2018). Creating and hosting websites on GitHub pages can be time-consuming, as this process usually requires more knowledge in web design than out-of-the-box platforms (*e.g.*, Wix, Weebly, Google Sites). However, free hosting, widely available template customization, and versioning are strong advantages over the alternatives.

Archiving citable code and data

Government, funding agencies, and publishers exercise rigorous open access data policies and mandates ([Nugroho et al., 2015](#); [Tenopir et al., 2020](#)). However, code and data sharing may be met by individual reluctance, temporary embargoes, or partially prevented by privacy and confidentiality reasons ([Figueiredo, 2017](#); [Tenopir et al., 2015](#); [Wicherts et al., 2011](#)). Still, data deposition and ensuring its availability can amplify the outreach of published studies ([Pronk et al., 2015](#)), increase citation rates ([Piwowar et al., 2007](#)), and among many other reasons, enable the reproducibility and robustness of scientific advances ([Baker, 2016](#); [Mislan et al., 2016](#); ["On Data Availability, Reproducibility and Reuse," 2017](#)). While public repositories on GitHub make it easy to store and share data files, they are not considered long-term repositories for research materials. This is because GitHub, a for-profit company, does not have long-term data availability guarantees, allowing users to delete or make repositories private after publication. Also, GitHub does not issue Digital Object Identifiers (DOI) for content uploaded to their servers. DOIs are persistent and citable unique alpha-numeric identifiers assigned to digitally stored research materials. Because of this, scientists sharing code and data through GitHub are strongly encouraged to independently submit their research materials to long-term data archives (*e.g.*, Zenodo, Figshare, Dryad, OSF ([Crystal-Ornelas et al., 2021](#); [Perez-Riverol et al., 2016](#); [J. Perkel, 2016](#)); [Table 1](#)). Some of these options (Zenodo, Figshare and OSF) integrate with GitHub, allowing project, code, and data releases ([Box 1](#)) to be archived with versioned, citable DOIs. Linking GitHub repositories with a DOI helps research become findable, properly cited, and can ensure long-term stability ([Hampton et al., 2015](#)). This strategy has been increasingly adopted in numerous studies in ecology and evolution (*e.g.*, the Zenodo repositores for Hudgins ([2022](#)), EllenJCoombs ([2020](#)), and Lrharper1 ([2018](#))).

An important aspect of making code and data citable and reusable is to add an appropriate license to protect intellectual property. Code published without a license is under exclusive copyright (by default), protecting it from copy, distribution, and modifications. One may grant specific rights to their code for reuse by adding licensing files and specifications within GitHub repositories (*[Adding a License to a Repository,](#)* n.d.). The Choose a License (https://choosealicense.com/non-software/) website offers further guidance on the licenses available for research and creative products. For example, Creative Commons (CC; https://creativecommons.org/licenses/) licenses can specify that shared code is intended for a specific analysis. A CC BY 4.0 license specifies that any code (or other creative products) must be appropriately credited to its original author when distributed, adapted or reused.

Collaborative and asynchronous code editing

Researchers can asynchronously communicate and individually or collaboratively work in GitHub. By forking, branching, and cloning repositories ([Box 1](#)), researchers can simultaneously work on different issues of the same research project to then later merge changes to the main branch project with pull requests. Explicit project organization and increased communication in GitHub Issues, Discussions, or within pull requests can help with project development and with potential merge conflicts due to users simultaneously working on the same sections ([Vale et al., 2019](#)). Moreover, version control ([Crystal-Ornelas et al., 2022](#)) and commit reversal features allow researchers to track and progress without worrying about irreparably modifying someone else's work. By enabling more comprehensive remote collaboration, GitHub encourages the exchange of ideas among researchers at different institutions and in different countries, which can serve to improve the quality of the research itself by providing open access to data and code. In academic settings, GitHub can also facilitate interactions between research advisors and advisees, providing a platform for students or other trainees to share in-progress code, and flag specific challenges or questions for their supervisors or mentors ([Table 2](#)). Periodic code review ([Song et al., 2020](#)) can also help advisors identify errors early in the process, and inform further training and mentorship to fill gaps in skills.

Writing a manuscript

Beyond supporting collaborative code development, GitHub can be used for writing manuscripts. Writing a manuscript and storing its associated data and code in GitHub increases scientific reproducibility because text, code, and data can be found in one place. Writing a manuscript on GitHub may take more time when compared to using conventional text processors ([Ram, 2013](#)). Nevertheless, GitHub has many features that can allow for a powerful collaborative workflow when writing manuscripts. Text documents stored and versioned in GitHub can be instantly displayed when written in Markdown, a lightweight markup language increasingly popular among scientists. Co-authors can contribute changes or suggest revisions to a manuscript written in GitHub through pull requests ([Box 1](#)). Pull requests provide line-by-line views of proposed changes, which can be commented, modified, or approved by designated reviewers and collaborators ([Table 2](#)). Relevant literature or issues can be made using the Discussions and Issues features. Moreover, real-time collaboration on text documents stored in GitHub repositories can be achieved with the help of other platforms [*e.g.*, HackMD (https://hackmd.io) for Markdown documents].

We wrote this manuscript using Manubot (*[Manubot - Manuscripts, Open and Automated,](#)* n.d.), a modifiable workflow implemented in GitHub to automatically render manuscripts and automate bibliographical tasks ([Himmelstein et al., 2019](#)). Manubot uses GitHub's automation workflow, GitHub Actions, to combine and convert individual Markdown files into a single LaTeX document, which can then be converted to a Word or PDF document, and displayed as a webpage. Citations and bibliographic references are automatically managed with citable persistent identifiers (*e.g.*, DOIs, PubMed IDs, ISBNs, URLs). The resulting manuscript can be rendered with document templates and citation style language formatting to meet journal formatting requirements. Every change made to the manuscript triggers its rendering, so that updates are readily displayed and made publicly available. Additional GitHub Actions can be integrated with Manubot, such as ones creating figures or generating tables (*e.g.*, https://github.com/SORTEE-Github-Hackathon/manuscript/tree/main/.github/workflows).

Peer review

Peer review is the standard process for assessing and judging whether research done in ecology and evolution should be published in a scientific journal. GitHub provides an open and transparent platform that can be used for either directly providing feedback on research products or addressing changes recommended by reviewers. GitHub Issues can be used to organize and discuss reviewer suggestions and to assign them to co-authors (*e.g.*, https://github.com/SORTEE-Github-Hackathon/manuscript/issues?q=label%3A%22Reviewer+Comment%22+). When reviewer comments are posted as separate issues, authors can comment on the issues to discuss possible changes and assign themselves to indicate which comments they intend to handle. Co-authors can then integrate their edits and responses to reviewers using pull requests.

GitHub can also assist reviewers during the peer review process. If the code associated with a manuscript is made available at the time of submission (*e.g.*, as a link to a GitHub repository within the Data Availability Statement), peer reviewers may be able to offer more comprehensive suggestions on the code and written materials, potentially recognizing errors before publication. Certain journals or software development communities require submitted work or research code to be hosted on GitHub and their review processes make use of GitHub Issues (*e.g.*, rOpenSci (https://ropensci.org/software-review/), Journal of Open Source Software (https://joss.readthedocs.io/en/latest/submitting.html)). rOpenSci's efforts have resulted in many well-used R packages for ecology research including `rfishbase` ([Boettiger et al., 2012](#)) and `taxize` ([Chamberlain & Szöcs, 2013](#)).

Open science discussion

Scientific publications often omit part of their intellectual and computational workflows, including the treatment of raw data and analytical steps (*e.g.*, model assumption testing). Publishing data and reproducible workflows along with manuscripts can provide readers with all details about analytical steps and enable reproducing research experiments and results ([Culina et al., 2020](#)). In addition to storing data and code, GitHub repositories can provide a time-stamped (version controlled) preregistration of research plans and hypotheses.

Conventional research practices typically separate tasks among collaborators (*i.e.*, data entry, analysis, writing). It is common that coauthors discuss, but do not actively verify, edit, or execute research tasks that are not their main responsibility. GitHub can serve as a tool for open and tractable research development. Collaborators can directly interact with code and data, inspect for errors, and potentially identify scientific misconduct prior to manuscript submission (*e.g.*, https://ecologyforthemasses.com/2020/02/04/pruittdata-and-the-ethics-of-data-in-science). Collaborators and readers are better positioned to discover erroneous or questionable findings if they have complete and transparent access to projects.

This transparency can be extended beyond co-authors to the entire scientific community and to the public. Supplying code for (novel) methods that are proposed or used reduces barriers to knowledge, improving the ability of others to build on existing work. This results in greater proliferation and accessibility for a broader audience. Projects can make use of GitHub Discussions (https://docs.github.com/en/discussions) to communicate among repository members (collaborators) and to engage with other scientists and the general public. Moreover, researchers can also use the GitHub Community (https://github.community/) forum to share expertise or request help from others on their analyses and ideas ([Table 2](#)).

Project continuity

Projects in ecology and evolution often involve graduate students, research assistants, and post-doctoral fellows, who hold limited-term positions ([Fehr et al., 2021](#)). Without clear plans on project continuity, the research code and data management upkeep tends to fall off as researchers move on to new projects or to other institutions. Additionally, code and data can be difficult to access when kept only on personal devices ([Vines et al., 2014](#)).

GitHub can facilitate project continuity among research code by making code handover easier ([Fehr et al., 2021](#); [Ram, 2013](#)). Through version control, the history of code and data from projects in ecology and evolution can be tracked accessible to future laboratory members and collaborators ([Lowndes et al., 2017](#)). Repositories and organizations can have designated data and code owners [or more appropriate, stewards; see Hampton et al. ([2015](#)), *About Code Owners* ([n.d.](#))], who can also can change through time allowing for the transition of code between research cohorts. Other project collaborators can contribute to repository design and development, and their active involvement can both aid authors ability to act as guarantors of the project, and the clarity and reproducibility of the project for future users. In (Figure [1](#)), we highlight several elements of good repository structure, and the various ways that contributors may interact with them.

GitHub also allows users to describe and store information about software and code dependencies, ensuring that code can run using the same version of software as when it was initially developed. This can be achieved through the access of GitHub repositories containing the necessary releases of such software, or through the use of automated Github Actions with specific software versions.

Automation

Automation has a strong potential to expand the scale and pace of research in ecology and evolution (Keitt & Abelson, 2021). Automation frameworks can streamline many stages of the scientific process, including automated data collection and data validation (*e.g.*, Micheletti et al., 2021; Yenni et al., 2018), automated data analysis (*e.g.*, Beaulieu-Jones & Greene, 2017), automated archiving and deployment of data, code and reports (*e.g.*, this manuscript, White et al., 2018). In this context, small modifications to code and data can be frequently committed and automatically tested, as in continuous integration and continuous deployment practices (Meyer, 2014). This allows for early detection and correction of errors, potentially improving confidence in scientific development by minimizing software errors (see Soergel, 2015). In addition to increasing scientific rigor and confidence in ecological software (Scheller et al., 2010), automation can help advance more rapidly sharing ecological data and making sure the data are high quality (Dietze et al., 2018). Integrating automation workflows has been highly encouraged in areas of EEB, including predictive ecology (McIntire et al., 2022), long-term ecological studies (Ernest et al., 2018; Yenni et al., 2019), and management of species at risk information (Naujokaitis-Lewis et al., 2022).

Automation can be integrated into GitHub repositories through the GitHub Actions feature (*Features • GitHub Actions*, n.d.), or through alternative automation systems (*e.g.*, Circle CI, *Continuous Integration and Delivery*, n.d.); Travis CI, *Homepage | Travis CI – Start Building Today!* (n.d.)]. Users can set up workflows associated with their repositories that are triggered by events (*e.g.*, push, pull request or at specified times) for remote servers to perform user-specified steps and actions. These actions are highly configurable and have numerous applications, such as automatically running analyses and creating figures when data or code are updated, incorporating changes to websites or applications, testing modifications to software (*e.g.*, R or Python libraries). Action workflows can be found in GitHub's Marketplace (https://github.com/marketplace?type=actions) or, alternatively, in open user repositories.

Organizing and managing teams

GitHub Organizations are shared virtual spaces that allow teams to work in different repositories, while remaining tied together under a larger group, such as a laboratory, department, or project involving several teams. Organizations allow larger projects with many steps or moving parts to be constrained to one virtual space, where outputs and sub-projects can be easily accessed and located without relying on individuals. Because the repositories are grouped, members can reference and contribute to each others work without necessarily being part of the same repository, broadening the accessibility and longevity of code and writing contributions.

Contributors can be assembled into teams within an organization, which allows administrators to assign roles, tasks, and repository modification permissions to organization members. Whereas access to repositories is usually assigned to individual contributors, Organizations facilitate the management of access permissions by allowing teams to be granted access to specific repositories. This ensures repositories with sensitive information remain as restricted as needed, while others stay open and accessible to selected member groups. The organization structure also allows for issue tracking and discussions related to research content and progress.

As an example, GitHub Organizations are particularly well-suited to host documents and projects within a laboratory, such as research compendia, codes of conduct, protocols, training documents, and other relevant documents that evolve collaboratively over time. In this way, teams have full ownership of repositories within an organization, while ensuring that these materials stay accessible to the laboratory after people have moved on or when locally-stored data are lost. This application extends to research centres, which may include several distinct projects that remain linked to institutions [*e.g.*, the German Centre for Integrative Biodiversity Research (iDiv, https://github.com/idiv-biodiversity)]. The team organizing the hackathon which inspired this article used a GitHub Organization (SORTEE-Github-Hackathon, https://github.com/SORTEE-Github-

[Hackathon](#)) to centralize the project development, from meeting notes to, ultimately, this manuscript. Organizations are also convenient for hosting learning materials, including lectures or workshops, such as the Québec Centre for Biodiversity Science R Workshop Series (QCBSRworkshops, https://github.com/QCBSRworkshops) or the University of Edinburgh's Coding Club (Coding Club, https://github.com/ourcodingclub), which may be continuously updated by an ever-evolving group of contributors.

Discussion

The promise of GitHub for EEB researchers

There have been many calls for researchers outside of the software development community to adopt GitHub to improve their collaborative research ([Anbaroğlu, 2021](#); [Crystal-Ornelas et al., 2021](#); [J. Perkel, 2016](#)). These calls come in light of the continuous shift towards open-science and the increasing computational and data requirements in EEB. Until now, resources and practical guidance on using GitHub within the EEB community have been dispersed in blog posts and video tutorials ([Box 2](#)). These resources have been useful for learning to use GitHub in our own work. We expect that situating the main uses of GitHub in EEB into one medium, while adding on our personal perspectives, will be useful to the EEB community.

The thirteen use cases we described here can leverage GitHub to enable more transparent and collaborative research in ecology and evolution (Figure [2](#)). Most of these uses can be quickly integrated into the research practices of users (*e.g.*, storing data, creating virtual notebooks, making code citable). Making stored data and code citable usually involves creating a repository on GitHub, pushing code and data, and then integrating a DOI minting service to the repository (*e.g.*, with Zenodo; see below). On the other hand, some examples we described here, including course material development, web hosting, and automation, require greater effort and time commitment, but have the potential to make EEB research more open, accessible, and collaborative. Managing full research projects or laboratories on GitHub require careful thought as to how to delegate tasks such as reviewing pull requests or creating issues. For example, collaboratively authoring a paper using GitHub, as we have done here, involves a learning curve for co-authors less familiar with the intricacies of GitHub, requires overhead to set up the automation framework through GitHub Actions, and especially, the commitment from all co-authors to use GitHub when modifying and reviewing the text. Despite the potential applications of GitHub to EEB research, we acknowledge that researchers might still look to other platforms for research collaboration.

Other platforms for collaboration

Despite its strong collaborative potential, we describe two use cases where GitHub's features fall short of highly collaborative work emblematic of EEB research.

First, real-time document editing is still best performed on other platforms (*e.g.*, cloud-stored documents from Microsoft Word, Google Docs, hackMD (https://hackmd.io/)). Second, operations that are dependent on other software might not be easily achievable in GitHub, such as manipulating figures or tables. Although creating tables and figures can be done through code, users may choose other software to collaboratively brainstorm figures and tables (*e.g.*, Google Slides, Google Sheets; but see GitHub Discussions).

Why aren't more EEB researchers using GitHub?

Although GitHub has been available as a platform for more than a decade, its uptake among EEB researchers, especially as a tool for collaboration, has been slow.

Here, we discuss four potential barriers to GitHub use in EEB.

First, there may be hesitation to independently adopting and learning a new tool. Institutional encouragement and instructional resources focused on researchers in ecology and evolution may be limited. When GitHub is taught within an EEB context, it usually accompanies coursework in topics such as statistical programming. It can be challenging to learn Git alongside scripting languages, statistical theory, and file system navigation, especially when many may be inexperienced with programming. Instructors likewise may confuse the expected digital literacy of students with computational fluency, even when modern technology increasingly abstracts concepts through search optimization and preponderant integrated development environments (IDE), or 'point-and-click' user interfaces.

Second, while EEB researchers individually use GitHub, collaborative use may lag due to researchers traditionally dividing labor within projects. Despite broad utility, GitHub remains a tool predominantly used by computer scientists and software developers. EEB researchers may take the view that GitHub is a platform that only needs to be used by individuals writing code and may silo those aspects of projects to a single individual. Those assumptions may obscure the utility of GitHub for tasks other than traditional data analysis and code development. However, we emphasize that there are opportunities for collaboration using GitHub by researchers of all skill levels or time constraints (Table 2); for example, project stakeholders can provide a list of use-cases or highlight important conceptual components of a project using GitHub Issues or Discussions features.

A third barrier may come from general reluctance to share data and code publicly, or technical and logistical issues (Gomes et al., 2022). GitHub is, by default, a public and open platform. This openness may add additional pressure to students and scientists learning to use the platform. File storage is size-limited on GitHub (but note that large files stored locally can still be versioned; see section "Storing and sharing research compendia"), requiring the complementation of other tools to fully integration project files and GitHub repositories (*e.g.*, (*Connect GitHub to a Project - OSF Support, n.d.*)). A major additional barrier to EEB researchers is the lack language-specific resources for non-English speaking researchers working in ecology and evolution. Language is a well-known obstacle to international collaborative research progress and to widespread scientific knowledge (see Khelifa et al., 2022). Non-English speaking EEB researchers can potentially miss opportunities to fully integrate version control, reproducibility, and other benefits of GitHub without language-inclusive contents.

Lastly, when projects become highly collaborative they may have to pay for GitHub features such as branch protections, multiple reviewers of pull requests, and time in its automation tools. Currently, GitHub offers Education Packs (https://education.github.com/) to students and academics, which extends some paid features to the free plan. However, the acquisition of GitHub by Microsoft has raised concerns over the future of free plans, causing several biodiversity data managers to shift to alternative Open Source Git services (*e.g.*, Bitbucket and GitLab).

Box 2

Ten tips for getting started in GitHub
1. **Check for existing solutions to your problem.** The GitHub Help webpage (https://docs.github.com/en) contains extensive and detailed documents with helpful screenshots. It is a good starting point for handling an issue, and has troubleshooting tips for specific problems. Alternatively, consider Tweeting your issue. There is a large community of GitHub users around the world who have likely faced analogous problems and may be able to provide quick solutions. Third, try to follow blogs (*e.g.*, https://github.blog), Twitter accounts or YouTube channels that regularly post practical solutions about the most widely-used web platform for common GitHub issues.

Ten tips for getting started in GitHub
2. **Consider taking free courses**, such as those from Software Carpentry ([Madicken Munk et al., 2019](#)), and sharing these courses with your lab members or colleagues.
3. **Take advantage of GitHub as an asynchronous working tool for team-based projects.** See the repository for this paper (https://github.com/SORTEE-Github-Hackathon/manuscript) as an example of a collaboratively authored manuscript that used the GitHub Discussions, Issues, Pages, and Actions features.
4. **Use the interactive courses from the GitHub Skills page (https://skills.github.com/)**, which allow you to learn GitHub basics through short projects and tasks with step-by-step guides.
5. **Learn markdown and use cheatsheets (*e.g.*, http://markdownguide.org/basic-syntax)** so you can write clear metadata README files for your repositories.
6. **Consult online resources.** The Jenny Bryan Universe of GitHub material, for example, provides a thorough and accessible introduction for a multitude of research-related uses for GitHub, and includes a book ([Hester, n.d.](#)), statistics course ([Bryan & TAs, n.d.](#)) and academic article ([Bryan, 2018](#)).
7. **Do not be afraid of trial-and-error.** One of the best ways to learn GitHub is the "trial-and-error" method. Learning from your own mistakes can be a valuable way to master your GitHub abilities. In any case, if you make mistakes, GitHub allows you to revert any steps that you desire via version controlling.
8. **If you are an educator, include lectures on reproducibility and tools for creating reproducible workflows in the curricula.** Some graduate programs include coursework on course R Markdown and GitHub. Getting students started with these tools earlier will prevent the resistance that comes from working with a less reproducible workflow for a longer period of time (see example https://github.com/rmcelreath/stat_rethinking_2022).
9. **Try to begin committing with graphical user interfaces (GUI) instead of command line interfaces (CLI).** Examples of GUI are the GitHub Desktop (https://desktop.github.com), git-gui (https://git-scm.com/docs/git-gui), RStudio (https://www.rstudio.com), Visual Studio Code (https://code.visualstudio.com), Atom (https://atom.io), GitKraken (https://www.gitkraken.com).
10. **Get help deciphering GitHub Notifications.** Try using tools like Octobox (https://octobox.io) to disentangle and manage multiple notifications from distinct GitHub projects.

Conclusion

We describe thirteen practical ways that ecologists and evolutionary biologists can adopt GitHub to improve their research workflow and make it more open and reproducible. We provide definitions ([Box 1](#)) and types of users (Figure [1](#)) to help researchers identify and prioritize the skills and tools to learn and apply. We highlight tools providing high collaborative potential (*e.g.*, open science discussion, collaborative code editing) to more individual focused (*e.g.*, storing code and data, building a website). We argue that the tools readily available in GitHub have the potential to make ecology and evolution more open, reproducible and transparent. With this comprehensive review of how EEB researchers can use GitHub, we encourage researchers at any career stage to adopt GitHub as a platform for sharing and collaboration.

Author Contributions

We indicate author contributions using the [CRediT Taxonomy](#).

- Conceptualization: RCO, BPME, KH, EJH, LLSR, PHPB
- Investigation: RCO, BPME, KH, EJH, LLSR, ERS, MJG, VF, ADB, CBB, KMG, SSHS, AG, FH, HW, DGEG, PHPB
- Methodology: RCO, BPME, KH, EJH, LLSR, PHPB
- Project administration: RCO
- Software: RCO, BPME, KH, EJH, LLSR, ERS, MJG, VF, ADB, CBB, KMG, SSHS, AG, FH, HW, DGEG, PHPB

- Visualization: RCO, BPME, KH, EJH, LLSR, ERS, MJG, VF, ADB, CBB, KMG, SSHS, AG, FH, HW, DGEG, PHPB
- Writing – original draft: RCO, BPME, KH, EJH, LLSR, ERS, MJG, VF, ADB, CBB, KMG, SSHS, AG, FH, HW, DGEG, PHPB
- Writing – review and editing: RCO, BPME, KH, EJH, LLSR, ERS, MJG, VF, ADB, CBB, KMG, SSHS, AG, FH, HW, DGEG, PHPB
- Resources: PHPB

Acknowledgements

Code and data availability

The source code and data for this manuscript are available at https://github.com/SORTEE-Github-Hackathon/manuscript. The source code will be archived in the ESS-DIVE data repository prior to publication.

Ethics declarations - Competing interests

RCO was recently offered a position to work at GitHub (on July 15, 2022) and will start employment at GitHub on August 23, 2022. Initial discussion of publishing this manuscript began in July 2021 and all work on the manuscript took place while RCO was an employee at Lawrence Berkeley National Laboratory.

Figures

Figure 1: An overview of Git's core features. A) Multi-faceted components allow for code writing, small data storage, manuscript writing, and project management to all be done in one place. `CONTRIBUTING.md`, `LICENCE.md`, and `README.md` files allow new team members, or others wanting to use materials, to understand the project components and learn how they can engage with the project and existing team members. B) Issues, Pull Requests, Discussions, and Projects allow for team members to ask for feedback, suggest fixes, discuss related ideas, and keep track of all the moving parts of a project. C) All collaborators on a project can be a part of a single repository, with varying push privileges and responsibilities
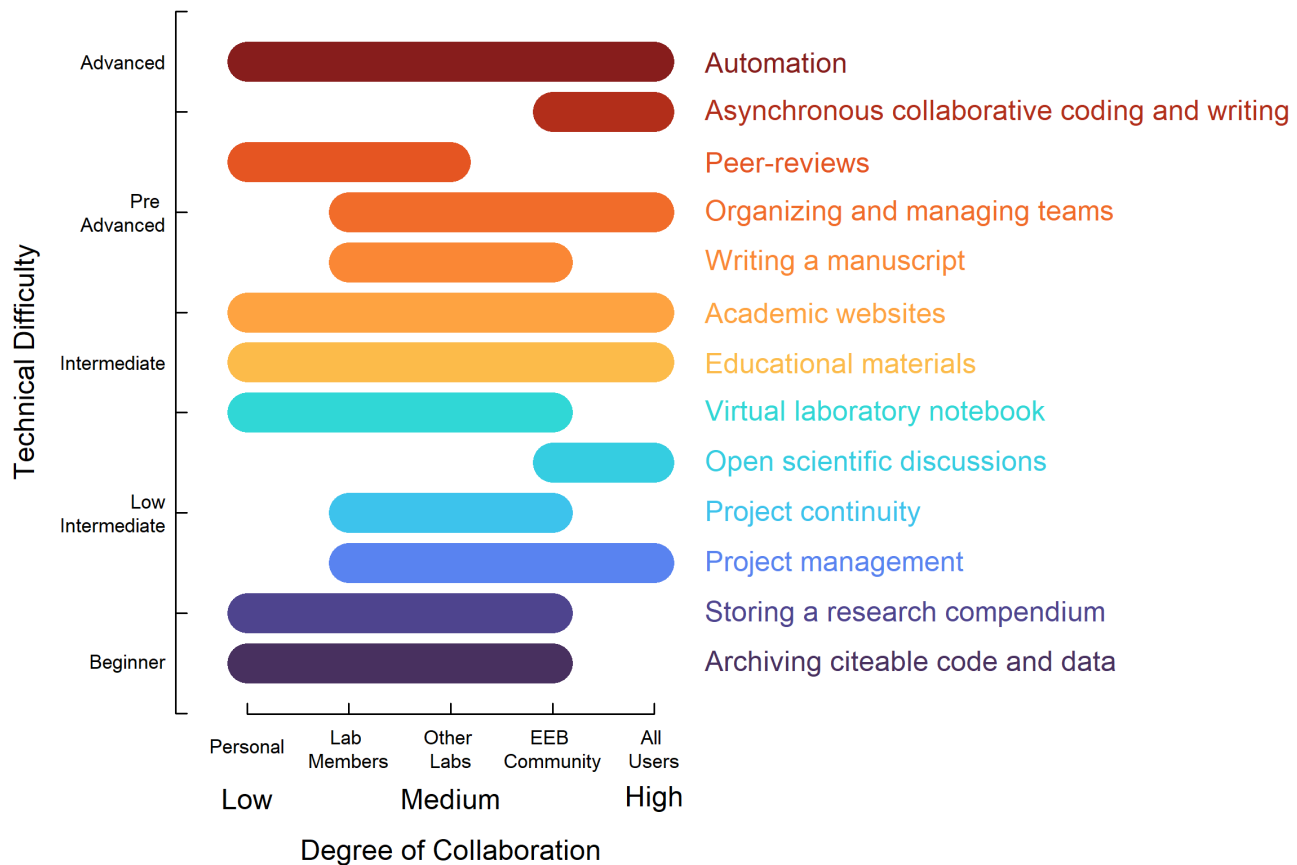
Figure 2: A summary of ways GitHub can be used showing technical difficulty and degree of collaboration for each. Activities higher on the vertical axis require usage knowledge of more GitHub features than activities lower on the axis. On the horizontal axis, each activity spans a region representing who is potentially involved with or benefits from each activity. For example, storing data and code mainly benefits individual researchers or members of a lab group while making data and code citable and reproducible benefit other labs and the larger community as well. Independently of a users knowledge level of GitHub features, there are ways to use GitHub that allow tapping unto one of the most salient benefits of the platform: facilitating and enhancing collaboration.

Tables

Table 1: A comparison of technologies commonly used for collaborating on research in Ecology and Evolutionary Biology. In the first column, we group platforms for collaboration into broad guilds. The second column lists the platform for collaboration. The remaining columns indicate whether the platform for collaboration includes certain features.

Guild	Software	Version control	Backup (cloud)	Passive collaboration	Active real-time collaboration	Free plan available	Permanent (DOI)	Storage limits	GitHub Integration
Multi-tool	GitHub	yes	yes	yes	no	Broadly used free version. Advanced features are provided for free to students and education professionals.	A DOI can only be obtained when integrating to other services that can mint DOI (e.g., Zenodo, OSF).	100MB per file, 500MB per private repository (2GB for paid accounts). 100GB for public repositories. Larger files (up to 2GB) can be attached to releases	N/A
Multi-tool	Open Science Framework	yes	yes	yes	yes	yes	yes	25GB for private projects, up to 5GB per file, plus partner add-ons, 50GB for public projects	yes
Long-term (public) data repositories	PANGAEA	yes	yes	yes	no	yes	yes	10 GB free	no
Long-term (public) data repositories	Zenodo	after published	after published	yes	no	yes	yes	50 GB per dataset	yes

Category	Service								
Long-term (public) data repositories	Dryad	after published	after published	yes	no	some journals cover cost	yes	300 GB per publication	Can link to individual files (not entire repository), thus not fully integrated
Long-term (public) data repositories	Figshare	yes	yes	yes	no	yes	yes	20 GB free, up to 5 TB	yes
Temporary (personal) drive storage	Google Drive	yes	yes	yes	yes	limited free version & paid	no	15GB free, up to 100GB with Google One	yes
Temporary (personal) drive storage	Box	limited	yes	yes	yes	no	no	Unlimited total size for subscription	yes
Temporary (personal) drive storage	DropBox	limited	yes	yes	yes	limited free version & paid	no	2GB free	yes
Temporary (personal) drive storage	One Drive and the Office Suite	yes	yes	yes	yes	limited free version & paid	no	5 GB free, up to 1TB paid	yes

Category	Tool									
Collaborative code/text editors	Overleaf (online latex editor)	yes	yes	yes	yes	yes		no	1MB for individual .tex, 50MB for individual files, unlimited project size	yes
Collaborative code/text editors	Jupyter Notebook	yes	?	yes	with Colab	yes		no	via Binder: no hard limit, but suggests no files >100MB, can also store on GitHub or Google Colab	yes
Collaborative code/text editors	HackMD	yes	yes	yes	yes	limited free version & paid		no	3 documents free, private invitee limits	yes

Table 2: A non-exhaustive collection of ideas for how various GitHub features could be utilized for a research project. Here we have categorized contributors/collaborators into five roles. A Project Manager owns the GitHub repository for a project, and leads the academic project (e.g., lead author of a manuscript). A co-author contributes to writing and other aspects of research, but may have limited or no experience with programming, git, and/or GitHub. A code contributor writes or edits analysis code for the project. A code reviewer could be a project collaborator or a peer reviewer who reviews project code. They are familiar with coding, but not necessarily with git or GitHub (but they are willing to learn). Finally, community members could be other researchers or non-researchers interested in reproducing results, re-using code or data, or communicating with researchers involved in the project. These roles are not mutually exclusive—a co-author could also be e code contributor and code reviewer, for example. For definitions of the GitHub features, see Box 1.

Role	GitHub repository	README	Issue	Discussion	Pull Request	Fork	GitHub Pages
Project manager	Set contributor permissions, share code of conduct	Project description, citation, DOIs	Assign tasks to collaborators	Discuss project directions and goals	Approve and incorporate edits to code and/or writing		Share up-to-date reports, figures, or draft manuscript
Co-author	Edit Markdown text or add files		Propose changes involving code (e.g. analyses, figures)	Discuss proposed changes to manuscript			
Code contributor			Suggest code changes		Contribute changes to code, initiate code review		Contribute to project website

Role	GitHub repository	README	Issue	Discussion	Pull Request	Fork	GitHub Pages
Code reviewer	Find all code related to a project		Highlight specific lines of code and make suggestions		Review or recommended changes in code		
Community			Suggest additional features and report bugs	Ask questions about data and code		Create a linked, editable copy of the repository	View project website

References

About code owners. (n.d.). GitHub Docs. Retrieved November 17, 2022, from https://ghdocs-prod.azurewebsites.net/en/repositories/managing-your-repositorys-settings-and-features/customizing-your-repository/about-code-owners

About large files on GitHub. (n.d.). GitHub Docs. Retrieved November 17, 2022, from https://ghdocs-prod.azurewebsites.net/en/repositories/working-with-files/managing-large-files/about-large-files-on-github

About Projects. (n.d.). GitHub Docs. Retrieved November 17, 2022, from https://ghdocs-prod.azurewebsites.net/en/issues/planning-and-tracking-with-projects/learning-about-projects/about-projects

Adding a license to a repository. (n.d.). GitHub Docs. Retrieved November 17, 2022, from https://ghdocs-prod.azurewebsites.net/en/communities/setting-up-your-project-for-healthy-contributions/adding-a-license-to-a-repository

Alston, J. M., & Rick, J. A. (2021). A Beginner's Guide to Conducting Reproducible Research. *The Bulletin of the Ecological Society of America*, *102*(2). https://doi.org/10.1002/bes2.1801

Anbaroğlu, B. (2021). A collaborative GIS programming course using GitHub Classroom. *Transactions in GIS*, *25*(6), 3132–3158. https://doi.org/10.1111/tgis.12810

Baker, M. (2016). 1,500 scientists lift the lid on reproducibility. *Nature*, *533*(7604), 452–454. https://doi.org/10.1038/533452a

Beaulieu-Jones, B. K., & Greene, C. S. (2017). Reproducibility of computational workflows is automated using continuous analysis. *Nature Biotechnology*, *35*(4), 342–346. https://doi.org/10.1038/nbt.3780

Boettiger, C., Lang, D. T., & Wainwright, P. C. (2012). rfishbase: exploring, manipulating and visualizing FishBase data from R. *Journal of Fish Biology*, *81*(6), 2030–2039. https://doi.org/10.1111/j.1095-8649.2012.03464.x

Briney, K., Coates, H., & Goben, A. (2020). Foundational Practices of Research Data Management. *Research Ideas and Outcomes*, *6*. https://doi.org/10.3897/rio.6.e56508

Bryan, J. (2018). Excuse Me, Do You Have a Moment to Talk About Version Control? *The American Statistician*, *72*(1), 20–27. https://doi.org/10.1080/00031305.2017.1399928

Bryan, J., & TAs, T. S. 545. (n.d.). *STAT 545*. Retrieved November 17, 2022, from https://stat545.com/

Build software better, together. (n.d.). GitHub. Retrieved November 17, 2022, from https://github.com

Chamberlain, S. A., & Szöcs, E. (2013). taxize: taxonomic search and retrieval in R. *F1000Research*, *2*, 191. https://doi.org/10.12688/f1000research.2-191.v2

Connect GitHub to a Project - OSF Support. (n.d.). Retrieved November 17, 2022, from https://help.osf.io/article/211-connect-github-to-a-project

Continuous Integration and Delivery. (n.d.). CircleCI. Retrieved November 17, 2022, from https://circleci.com/

Crystal-Ornelas, R., Edwards, B. P. M., Hébert, K., Hudgins, E. J., Reyes, L. L. S., Scott, E. R., Grainger, M. J., Foroughirad, V., Binley, A. D., Brookson, C. B., Gaynor, K. M., Sabet, S. S., Güncan, A., Hillemann,

F., Weierbach, H., Gomes, D. G. E., & Braga, P. H. P. (2022). *Not just for programmers: How GitHub can accelerate collaborative and reproducible research in ecology and evolution*. Manubot. https://SORTEE-Github-Hackathon.github.io/manuscript/

Crystal-Ornelas, R., Varadharajan, C., Bond-Lamberty, B., Boye, K., Burrus, M., Cholia, S., Crow, M., Damerow, J., Devarakonda, R., Ely, K. S., Goldman, A., Heinz, S., Hendrix, V., Kakalia, Z., Pennington, S. C., Robles, E., Rogers, A., Simmonds, M., Velliquette, T., … Agarwal, D. A. (2021). A Guide to Using GitHub for Developing and Versioning Data Standards and Reporting Formats. *Earth and Space Science*, *8*(8). https://doi.org/10.1029/2021ea001797

Culina, A., van den Berg, I., Evans, S., & Sánchez-Tójar, A. (2020). Low availability of code in ecology: A call for urgent action. *PLOS Biology*, *18*(7), e3000763. https://doi.org/10.1371/journal.pbio.3000763

Dawson, C. (2016). *Building tools with GitHub: customize your workflow* (First Edition). O'Reilly.

Dietze, M. C., Fox, A., Beck-Johnson, L. M., Betancourt, J. L., Hooten, M. B., Jarnevich, C. S., Keitt, T. H., Kenney, M. A., Laney, C. M., Larsen, L. G., Loescher, H. W., Lunch, C. K., Pijanowski, B. C., Randerson, J. T., Read, E. K., Tredennick, A. T., Vargas, R., Weathers, K. C., & White, E. P. (2018). Iterative near-term ecological forecasting: Needs, opportunities, and challenges. *Proceedings of the National Academy of Sciences*, *115*(7), 1424–1432. https://doi.org/10.1073/pnas.1710231115

EllenJCoombs. (2020). *EllenJCoombs/Asymmetry-evolution-cetaceans-for-publication-v.1.0.0* (Version v.1.0.0) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.3893943

Ernest, S. K. M., Yenni, G. M., Allington, G., Bledsoe, E. K., Christensen, E. M., Diaz, R. M., Geluso, K., Goheen, J. R., Guo, Q., Heske, E., Kelt, D., Meiners, J. M., Munger, J., Restrepo, C., Samson, D. A., Schutzenhofer, M. R., Skupski, M., Supp, S. R., Thibault, K., … Valone, T. J. (2018). *The Portal Project: a long-term study of a Chihuahuan desert ecosystem*. Cold Spring Harbor Laboratory. https://doi.org/10.1101/332783

Features • GitHub Actions. (n.d.). GitHub. Retrieved November 17, 2022, from https://github.com/features/actions

Fehr, J., Himpe, C., Rave, S., & Saak, J. (2021). Sustainable Research Software Hand-Over. *Journal of Open Research Software*, *9*(1), 5. https://doi.org/10.5334/jors.307

Figueiredo, A. S. (2017). Data Sharing: Convert Challenges into Opportunities. *Frontiers in Public Health*, *5*. https://doi.org/10.3389/fpubh.2017.00327

Gomes, D. G. E., Pottier, P., Crystal-Ornelas, R., Hudgins, E. J., Foroughirad, V., Sánchez-Reyes, L. L., Turba, R., Martinez, P. A., Moreau, D., Bertram, M., Smout, C., & Gaynor, K. (2022). *Why don't we share data and code? Perceived barriers and benefits to public archiving practices*. Center for Open Science. https://doi.org/10.31222/osf.io/gaj43

Goring, S. J., Weathers, K. C., Dodds, W. K., Soranno, P. A., Sweet, L. C., Cheruvelil, K. S., Kominoski, J. S., Rüegg, J., Thorn, A. M., & Utz, R. M. (2014). Improving the culture of interdisciplinary collaboration in ecology by expanding measures of success. *Frontiers in Ecology and the Environment*, *12*(1), 39–47. https://doi.org/10.1890/120370

Hampton, S. E., Anderson, S. S., Bagby, S. C., Gries, C., Han, X., Hart, E. M., Jones, M. B., Lenhardt, W. C., MacDonald, A., Michener, W. K., Mudge, J., Pourmokhtarian, A., Schildhauer, M. P., Woo, K. H., & Zimmerman, N. (2015). The Tao of open science for ecology. *Ecosphere*, *6*(7), art120. https://doi.org/10.1890/es14-00402.1

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

Hester, J. B., the STAT 545 TAs, Jim. (n.d.). *Let's Git started | Happy Git and GitHub for the useR*. Retrieved November 17, 2022, from https://happygitwithr.com/

Himmelstein, D. S., Rubinetti, V., Slochower, D. R., Hu, D., Malladi, V. S., Greene, C. S., & Gitter, A. (2019). Open collaborative writing with Manubot. *PLOS Computational Biology*, *15*(6), e1007128. https://doi.org/10.1371/journal.pcbi.1007128

Homepage | Travis CI – Start building today! (n.d.). Travis CI. Retrieved November 17, 2022, from https://www.travis-ci.com/

Hudgins, E. (2022). *emmajhudgins/UStreedamage: publication release* (accepted) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.6097109

Issues · fmsabatini/sPlotOpen_Manuscript. (n.d.). GitHub. Retrieved November 17, 2022, from https://github.com/fmsabatini/sPlotOpen_Manuscript

Kanza, S., Willoughby, C., Gibbins, N., Whitby, R., Frey, J. G., Erjavec, J., Zupančič, K., Hren, M., & Kovač, K. (2017). Electronic lab notebooks: can they replace paper? *Journal of Cheminformatics*, *9*(1). https://doi.org/10.1186/s13321-017-0221-3

Keitt, T. H., & Abelson, E. S. (2021). Ecology in the age of automation. *Science*, *373*(6557), 858–859. https://doi.org/10.1126/science.abi4692

Khelifa, R., Amano, T., & Nuñez, M. A. (2022). A solution for breaking the language barrier. *Trends in Ecology &Amp; Evolution*, *37*(2), 109–112. https://doi.org/10.1016/j.tree.2021.11.003

Kim, A. Y., Herrmann, V., Barreto, R., Calkins, B., Gonzalez-Akre, E., Johnson, D. J., Jordan, J. A., Magee, L., McGregor, I. R., Montero, N., Novak, K., Rogers, T., Shue, J., & Anderson-Teixeira, K. J. (2022). Implementing GitHub Actions continuous integration to reduce error rates in ecological data collection. *Methods in Ecology and Evolution*, *13*(11), 2572–2585. https://doi.org/10.1111/2041-210x.13982

Leibzon, W. (2016, August). Social network of software development at GitHub. *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM). https://doi.org/10.1109/asonam.2016.7752419

Lowndes, J. S. S., Best, B. D., Scarborough, C., Afflerbach, J. C., Frazier, M. R., O'Hara, C. C., Jiang, N., & Halpern, B. S. (2017). Our path to better science in less time using open data science tools. *Nature Ecology &Amp; Evolution*, *1*(6). https://doi.org/10.1038/s41559-017-0160

Lrharper1. (2018). *Hulluni-Bioinformatics/Harper_Et_Al_2018: Github Repository Deposited With Harper Et Al. (2018) In Ecology And Evolution*. Zenodo. https://doi.org/10.5281/zenodo.1188710

Madicken Munk, Koziar, K., Leinweber, K., Raniere Silva, Michonneau, F., McCue, R., Hejazi, N., Waldman, S., Emonet, R., Harris, R. M., Olex, A. L., Becker, E. A., Lever, J., Burle, M.-H., Moore, B., Umihiko Hoshijima, Maji, A., Topçuoğlu, B. D., Junghans, C., … Wolmar Nyberg Åkerström. (2019). *swcarpentry/git-novice: Software Carpentry: Version Control with Git, June 2019* (Version v2019.06.1). Zenodo. https://doi.org/10.5281/zenodo.3264950

Manubot - Manuscripts, open and automated. (n.d.). Retrieved November 17, 2022, from
    https://manubot.org

Marwick, B., Boettiger, C., & Mullen, L. (2018). Packaging Data Analytical Work Reproducibly Using R
    (and Friends). *The American Statistician*, *72*(1), 80–88.
    https://doi.org/10.1080/00031305.2017.1375986

McIntire, E. J. B., Chubaty, A. M., Cumming, S. G., Andison, D., Barros, C., Boisvenue, C., Haché, S., Luo,
    Y., Micheletti, T., & Stewart, F. E. C. (2022). PERFICT: A Re-imagined foundation for predictive
    ecology. *Ecology Letters*, *25*(6), 1345–1351. https://doi.org/10.1111/ele.13994

Meyer, M. (2014). Continuous Integration and Its Tools. *IEEE Software*, *31*(3), 14–16.
    https://doi.org/10.1109/ms.2014.58

Micheletti, T., Stewart, F. E. C., Cumming, S. G., Haché, S., Stralberg, D., Tremblay, J. A., Barros, C., Eddy,
    I. M. S., Chubaty, A. M., Leblond, M., Pankratz, R. F., Mahon, C. L., Van Wilgenburg, S. L., Bayne, E.
    M., Schmiegelow, F., & McIntire, E. J. B. (2021). Assessing Pathways of Climate Change Effects in
    SpaDES: An Application to Boreal Landbirds of Northwest Territories Canada. *Frontiers in Ecology
    and Evolution*, *9*. https://doi.org/10.3389/fevo.2021.679673

Mislan, K. A. S., Heer, J. M., & White, E. P. (2016). Elevating The Status of Code in Ecology. *Trends in
    Ecology &Amp; Evolution*, *31*(1), 4–7. https://doi.org/10.1016/j.tree.2015.11.006

Naujokaitis-Lewis, I., Endicott, S., & Guezen, J. M. (2022). CAN-SAR: A database of Canadian species at
    risk information. *Scientific Data*, *9*(1). https://doi.org/10.1038/s41597-022-01381-8

Nugroho, R. P., Zuiderwijk, A., Janssen, M., & de Jong, M. (2015). A comparison of national open data
    policies: lessons learned. *Transforming Government: People, Process and Policy*, *9*(3), 286–308.
    https://doi.org/10.1108/tg-03-2014-0008

On data availability, reproducibility and reuse. (2017). *Nature Cell Biology*, *19*(4), 259–259.
    https://doi.org/10.1038/ncb3506

Perez-Riverol, Y., Gatto, L., Wang, R., Sachsenberg, T., Uszkoreit, J., Leprevost, F. da V., Fufezan, C.,
    Ternent, T., Eglen, S. J., Katz, D. S., Pollard, T. J., Konovalov, A., Flight, R. M., Blin, K., & Vizcaíno, J. A.
    (2016). Ten Simple Rules for Taking Advantage of Git and GitHub. *PLOS Computational Biology*,
    12(7), e1004947. https://doi.org/10.1371/journal.pcbi.1004947

Perkel, J. (2016). Democratic databases: science on GitHub. *Nature*, *538*(7623), 127–128.
    https://doi.org/10.1038/538127a

Perkel, J. M. (2020). Challenge to scientists: does your ten-year-old code still run? *Nature*, *584*(7822),
    656–658. https://doi.org/10.1038/d41586-020-02462-7

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. https://doi.org/10.1371/journal.pone.0000308

Prabhu, P., Zhang, Y., Ghosh, S., August, D. I., Huang, J., Beard, S., Kim, H., Oh, T., Jablin, T. B., Johnson,
    N. P., Zoufaly, M., Raman, A., Liu, F., & Walker, D. (2011). A survey of the practice of computational
    science. *State of the Practice Reports on - SC '11*. State of the Practice Reports.
    https://doi.org/10.1145/2063348.2063374

Pronk, T. E., Wiersma, P. H., van Weerden, A., & Schieving, F. (2015). A game theoretic analysis of
    research data sharing. *PeerJ*, *3*, e1242. https://doi.org/10.7717/peerj.1242

Ram, K. (2013). Git can facilitate greater reproducibility and increased transparency in science. *Source
    Code for Biology and Medicine*, *8*(1). https://doi.org/10.1186/1751-0473-8-7

Scheller, R. M., Sturtevant, B. R., Gustafson, E. J., Ward, B. C., & Mladenoff, D. J. (2010). Increasing the reliability of ecological models using modern software engineering techniques. *Frontiers in Ecology and the Environment*, *8*(5), 253–260. https://doi.org/10.1890/080141

Schnell, S. (2015). Ten Simple Rules for a Computational Biologist's Laboratory Notebook. *PLOS Computational Biology*, *11*(9), e1004385. https://doi.org/10.1371/journal.pcbi.1004385

Slack. (n.d.). *GitHub for Slack*. Slack Help Center. Retrieved November 17, 2022, from https://slack.com/help/articles/232289568-GitHub-for-Slack

Smaglik, P. (2007). Creating better lab websites gives potential collaborators and recruiters a clearer window into your world. *Nature*, *447*(7142), 347–347. https://doi.org/10.1038/nj7142-347a

Soergel, D. A. W. (2015). Rampant software errors may undermine scientific results. *F1000Research*, *3*, 303. https://doi.org/10.12688/f1000research.5930.2

Song, X., Goldstein, S. C., & Sakr, M. (2020, June 2). Using Peer Code Review as an Educational Tool. *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education*. ITiCSE '20: Innovation and Technology in Computer Science Education. https://doi.org/10.1145/3341525.3387370

Tenopir, C., Dalton, E. D., Allard, S., Frame, M., Pjesivac, I., Birch, B., Pollock, D., & Dorsett, K. (2015). Changes in Data Sharing and Data Reuse Practices and Perceptions among Scientists Worldwide. *PLOS ONE*, *10*(8), e0134826. https://doi.org/10.1371/journal.pone.0134826

Tenopir, C., Rice, N. M., Allard, S., Baird, L., Borycz, J., Christian, L., Grant, B., Olendorf, R., & Sandusky, R. J. (2020). Data sharing, management, use, and reuse: Practices and perceptions of scientists worldwide. *PLOS ONE*, *15*(3), e0229003. https://doi.org/10.1371/journal.pone.0229003

Trujillo, G., & Tanner, K. D. (2014). Considering the Role of Affect in Learning: Monitoring Students' Self-Efficacy, Sense of Belonging, and Science Identity. *CBE—Life Sciences Education*, *13*(1), 6–15. https://doi.org/10.1187/cbe.13-12-0241

Vale, G., Schmid, A., Santos, A. R., de Almeida, E. S., & Apel, S. (2019). On the relation between Github communication activity and merge conflicts. *Empirical Software Engineering*, *25*(1), 402–433. https://doi.org/10.1007/s10664-019-09774-x

Vines, Timothy H., Albert, Arianne Y. K., Andrew, Rose L., Débarre, F., Bock, Dan G., Franklin, Michelle T., Gilbert, Kimberly J., Moore, J.-S., Renaut, S., & Rennison, Diana J. (2014). The Availability of Research Data Declines Rapidly with Article Age. *Current Biology*, *24*(1), 94–97. https://doi.org/10.1016/j.cub.2013.11.014

White, E. P., Yenni, G. M., Taylor, S. D., Christensen, E. M., Bledsoe, E. K., Simonis, J. L., & Ernest, S. K. M. (2018). Developing an automated iterative near-term forecasting system for an ecological study. *Methods in Ecology and Evolution*, *10*(3), 332–344. https://doi.org/10.1111/2041-210x.13104

Wicherts, J. M., Bakker, M., & Molenaar, D. (2011). Willingness to Share Research Data Is Related to the Strength of the Evidence and the Quality of Reporting of Statistical Results. *PLoS ONE*, *6*(11), e26828. https://doi.org/10.1371/journal.pone.0026828

Wilson, G., Aruliah, D. A., Brown, C. T., Chue Hong, N. P., Davis, M., Guy, R. T., Haddock, S. H. D., Huff, K. D., Mitchell, I. M., Plumbley, M. D., Waugh, B., White, E. P., & Wilson, P. (2014). Best Practices for Scientific Computing. *PLoS Biology*, *12*(1), e1001745. https://doi.org/10.1371/journal.pbio.1001745

Xie, Y., Allaire, J. J., & Grolemund, G. (2019). *R Markdown: the definitive guide*. CRC Press, Taylor and Francis Group.

Yenni, G. M., Christensen, E. M., Bledsoe, E. K., Supp, S. R., Diaz, R. M., White, E. P., & Ernest, S. K. M. (2019). Developing a modern data workflow for regularly updated data. *PLOS Biology*, *17*(1), e3000125. https://doi.org/10.1371/journal.pbio.3000125

Yenni, G. M., Christensen, E. M., Bledsoe, E. K., Supp, S. R., Diaz, R. M., White, E. P., & Ernest, S. K. M. (2018). *Developing a modern data workflow for evolving data*. Cold Spring Harbor Laboratory. https://doi.org/10.1101/344804