The SORTEE Guidelines for Data and Code Quality Control in Ecology and Evolutionary Biology

Joel L. Pick[1]*, Bethany J. Allen[2], Benedicte Bachelot[3], Kevin R. Bairos-Novak[4], Jack A. Brand[5,6], Barbara Class[7], Tad Dallas[8], Pietro B. D'Amelio[9], Erola Fenollosa[10], Esteban Fernández-Juricic[11], Dylan G. E. Gomes[12], Matthew J. Grainger[13], Thomas Guillemaud[14], Christian John[15], Ruby Krasnow[16], Malgorzata Lagisz[17,18], Sebastian Lequime[19], Daniel S. Maynard[20], Shinichi Nakagawa[18], Rose E. O'Dea[21], Matthieu Paquet[22], Quentin Petitjean[23], Alfredo Sánchez-Tójar[24], Natalie E. van Dis[25,26], Laura A. B. Wilson[17,27], Edward R. Ivimey Cook[28]*

*Corresponding authors: joel.l.pick@gmail.com and e.ivimeycook@googlemail.com

[1]Institute of Ecology and Evolution, University of Edinburgh, Edinburgh UK
[2]GFZ Helmholtz Centre for Geosciences, Potsdam, Germany
[3]Oklahoma State University, OK, USA
[4]Australian Institute of Marine Science, Townsville, Australia
[5]Department of Wildlife, Fish, and Environmental Studies, Swedish University of Agricultural Sciences, Umeå 907 36, Sweden
[6]Institute of Zoology, Zoological Society of London, London NW1 4RY, UK
[7]Direction pour la Science Ouverte (DipSO), INRAE, France
[8]University of South Carolina, SC, USA
[9]Department of Biology, Reed College, Portland, Oregon, USA
[10]Department of Biology, University of Oxford, UK
[11]Purdue University, USA
[12]Marine Reserves, Oregon Department of Fish and Wildlife, Newport, OR, 97365, USA
[13]Knowledge Synthesis Department, Norwegian Institute for Nature Research (NINA), Trondheim, Norway
[14]INRAE, Université Côte d'Azur, ISA, Sophia-Antipolis France
[15]Marine Science Institute, University of California, Santa Barbara. Santa Barbara, CA 93106 USA
[16]University of Maine, School of Marine Sciences, Orono, ME, USA

[17]Evolution & Ecology Research Centre, School of Biological, Earth & Environmental Sciences, University of New South Wales, Kensington, NSW, 2052, Australia
[18]Department of Biological Sciences, University of Alberta, Edmonton, AB T6G 2E9, Canada
[19]Cluster of Microbial Ecology, Groningen Institute for Evolutionary Life Sciences, University of Groningen, Groningen, The Netherlands
[20]Department of Genetics, Evolution, and Environment, University College London, London, UK
[21]School of Agriculture, Food and Ecosystem Sciences, University of Melbourne
[22]SETE, Station d'Écologie Théorique et Expérimentale, CNRS, Moulis, France
[23]Abeilles et Environnement (UR406), Institut National de Recherche pour l'Agriculture, l'Alimentation et l'Environnement (INRAE), Avignon, France.
[24]Department of Evolutionary Biology, Bielefeld University, Germany
[25]Organismal and Evolutionary Biology, University of Helsinki, P.O. Box 4, 00014 Helsinki, Finland
[26]Department of Animal Ecology, Netherlands Institute of Ecology (NIOO-KNAW), PO Box 50, 6700 AB Wageningen, The Netherlands
[27]School of Archaeology and Anthropology, The Australian National University, Acton ACT 2601, Australia
[28]School of Biodiversity, One Health, and Veterinary Medicine, University of Glasgow, Glasgow, UK

Abstract

Open data and code are crucial to increasing transparency and reproducibility, and in building trust in scientific research. However, despite an increasing number of journals in ecology and evolutionary biology mandating for data and code to be archived alongside published articles, the amount and quality of archived data and code, and subsequent reproducibility of results, has remained worryingly low. As a result, a handful of journals have recruited dedicated data editors, whose role is to help authors increase the overall quality of archived data and code. There is, however, a general lack of consensus of what a data editor should check, how to do it, and to what level of detail, and the process is often vague and hidden from readers and authors alike. Here, with the input from multiple data editors across several journals in ecology and evolutionary biology, we establish and describe the first standardised guidelines for Data and Code Quality Control on behalf of the Society for Open, Reliable, and Transparent Ecology and Evolutionary Biology (SORTEE). We start by introducing the concept of a data editor and data and code quality control, what is expected from data and code quality control, the relative costs and benefits to journals, authors, and readers, and then introduce and detail the SORTEE-led guidelines, ending with advice for journals and authors. We believe that by adopting these standardised guidelines, journals will help increase the consistency and transparency of the data editor process for readers, authors, and data editors.

Keywords: Data sharing, Code Sharing, Computational Reproducibility, Open Science, Data Re-use, Methodological Rigor, FAIR principles, Transparency, Data Editor

Introduction

A major focus of open science efforts in the last two decades, especially in ecology and evolutionary biology, has been open data, and more recently, open code. Open data and open code refer to the public archiving of the data and code associated with published research. The goals and societal benefits of data and code archiving have been widely discussed (Parr & Cummings, 2005; Barnes, 2010; Molloy, 2011; Wilkinson *et al.*, 2016; Goldacre *et al.*, 2019; Gomes *et al.*, 2022; Ivimey-Cook *et al.*, 2023; Box 1). Despite some reticence about open data and code (see Gomes *et al.*, 2022 for an overview of these fears), previous work has shown that data archiving is supported by the majority of academics in ecology and evolutionary biology who perceive that the benefits outweigh any costs (Soeharjono & Roche, 2021). Indeed, the two most important issues to the more than 800 members of the Society for Open, Reliable, and Transparent Ecology and Evolutionary Biology (SORTEE) from 2021-2024 have consistently been open data and open code (https://osf.io/gsw6x).

BOX 1: Goals of data and code archiving

From an idealistic perspective, data and code archiving has three main goals: to allow data reuse, to increase transparency, and to provide computational reproducibility (components that have been highlighted previously, e.g., Wilkinson *et al.*, 2016). We cover each of these in turn below.

A. Allow data reuse

The main focus of data archiving in the past has been on ensuring the potential for data reuse. Most prominently, this has included the development of the FAIR principles (Findable, Accessible, Interoperable, Reusable; Wilkinson *et al.*, 2016). There are several key motivations for this. First, data archiving prevents data loss. Data is typically collected using public money and so can be seen as a public good that should not be lost when a researcher leaves their job or when a computer is lost or broken. Loss of data (and code) represents a massive source of research waste (Purgar *et al.*, 2022). Second, archived data better allows for research synthesis (Culina *et al.*, 2020; Hennessy *et al.*, 2022). Meta-analysis plays a key role in generalising results across systems, however, published results are

often not described in enough detail to be included in a meta-analysis. Provision of the data allows re-analysis to gain the information a meta-analyst needs and can substantially increase the data available for synthesis (Kim *et al.*, 2021). Third, methods are frequently updated, and data archiving allows for data to be re-analysed when new methods become available. Finally, data archiving allows for new questions to be asked with existing data. This is not only an efficient use of time and money, but reduces the use of animals in experiments, a central tenet of animal ethics (Janssens *et al.*, 2023).

To achieve the goal of allowing data reuse, it is essential that data and their accompanying metadata (Table 1) are both present and in a form that allows for reuse. However, archived data are often incomplete or not in a state where they can be reused (Roche *et al.*, 2015, 2022). Helping authors to ensure their archived data are FAIR should therefore be a key goal for data editors.

B. Increase transparency

Another key goal of data and code archiving is to increase transparency. Transparency involves making the research process visible, and is associated with building trust and credibility in science (Vazire, 2017). While trust in scientists remains high globally, even a small minority's distrust can shape how research findings are received by the public and decision-makers (Cologna *et al.*, 2025), reinforcing the need for researchers to actively build and strengthen trust both with other academics and with the general public.

Analyses in ecology and evolutionary biology are becoming increasingly complex (Touchon & McCoy, 2016; Feng *et al.*, 2020), and there are often several ways to perform an analysis, with varying outcomes (Gould *et al.*, 2025). Descriptions of data filtering, processing, and analysis included within articles are not always sufficient to fully reproduce analyses (Archmiller *et al.*, 2020; Minocher *et al.*, 2021; Table 1). Provision of analysis code alongside a manuscript would therefore allow the analytical methods used to be directly assessed by reviewers when provided during peer review (Fernández-Juricic, 2021) and by the general readership upon publication. Transparent methods allow work to be built on more easily, making science more efficient. Unlike data, the goal of code archiving in empirical articles is rarely direct code reuse (if this is the goal, then it is often more appropriate to create software packages, to allow generalisation of a method). Code archiving instead allows software code to be used for reference, adapted for new use, to allow similar methods to be applied to new datasets, or for methods to be extended.

Data and code archiving also allow mistakes to be found. Coding mistakes are easily made, and whilst many may have negligible effects on the results of an article, some will have major effects (Gihawi *et al.*, 2023; Mandhane, 2024). The availability of the data and code that support an article make it possible for these mistakes to be found, and importantly, corrected in the future (Bolnick & Paull, 2016; Manzanedo *et al.*, 2021). Finally, following several high profile cases of academic fraud (e.g., Viglione, 2020, https://retractionwatch.com/2017/05/01/remarkable-ever-accepted-says-report-science-retract-study-fish-microplastics/, https://retractionwatch.com/2022/08/09/science-retracts-ocean-acidification-paper-more-than-a-year-after-a-report-on-allegations-in-its-own-pages/), it has become increasingly clear that, as a community, we would benefit from a higher degree of transparency in how the results of published articles are generated. Although the provision of data and code will not stop data fabrication, publicly available data makes detecting data fabrication much easier as the data can be scrutinised and presents an additional hurdle to generating fraudulent results. As data and code are increasingly provided alongside journal articles in our field (although more so for data than code; Culina *et al.*, 2020; Kimmel *et al.*, 2023; Sánchez-Tójar *et al.*, 2025) not providing these resources leads readers to ask, 'What are the reasons why the authors did not want to share their data and code?' (see Gomes *et al.*, 2022).

To fulfil the goal of transparency, readers therefore need to *see* what has been done to obtain the reported results. This requires the presence of all data and code needed to reproduce the results presented in an article, as well as linking what was done in that article with the structure and form of both the data and code (through the use of metadata and appropriate code annotation).

C. Provide computational reproducibility

Perhaps the most ambitious goal of data and code archiving is computational reproducibility, which ultimately builds trust and credibility in published results (Powers & Hampton, 2019; Reinecke *et al.*, 2022). We can define computational reproducibility as '*obtaining consistent results using the same input data, computational methods, and conditions of analysis*' (National Academies of Sciences *et al.*, 2019). Although this definition of computational reproducibility is commonly cited, the terms within it are not clearly defined. In the context of a research article, we interpret this to mean that, given the available data (i.e. input data) and code or workflow (i.e. computational methods), and using the same software versions (and hardware if appropriate) outlined in the article and metadata (i.e. conditions of analysis), we should be able to reproduce the results presented in the article.

In some cases, exact reproducibility is difficult to achieve (i.e. generating the *exact* numbers presented in an article), and often computational reproducibility is assessed with some tolerance level (e.g., (Archmiller *et al.*, 2020; Kambouris *et al.*, 2024). However, practices such as having appropriate metadata that adequately describes the software and package versions and setting of seeds (where the same pseudorandom numbers are generated each time) within code for stochastic methods, will help to achieve these goals. We should note that it is unlikely we will be able to demonstrate computational reproducibility in cases where analyses are highly computationally intensive, but solutions exist. We discuss these points in more detail in the main text.

To achieve the goal of computational reproducibility, as a minimum requirement, we need all code and data to be present. The next component of computational reproducibility is that the provided code runs without error. This requires the code to be explicit about what data files it uses and where these are located, it requires data files to be in the same directory structure and with the same names as expected by the code, and it requires the same version of all software packages used in the code to be loaded. Any code that cannot be rerun without error in a clean workspace cannot be considered computationally reproducible.

In response to this call for open data, starting in 2010, many journals in ecology and evolutionary biology began to mandate data archiving, e.g., Journal of Animal Ecology, Functional Ecology, and Heredity, to name a few (for a full list see the Joint Data Archiving Policy; https://datadryad.org/docs/JointDataArchivingPolicy.pdf). As a result, an increasing number of journals in ecology and evolutionary biology have mandatory open data policies (estimated to be 20% out of 196 journals in 2020, 35% in 2023; Berberi & Roche, 2022, 2023; and 41% in 2024; Ivimey-Cook *et al.*, 2025). This action has resulted in a large increase in the number of publications in ecology and evolutionary biology having open data (Vines *et al.*, 2013; Culina *et al.* 2020 found 79% in 14 journals that have a code archiving policy with no change from 2015/16 to 2018/19; Sánchez-Tójar *et al.*, 2025 found 37% in 12 journals without code archiving policies with an increase over time; Kimmel *et al.*, 2023 found 78.5% in 5 journals from 2018-20; Belkhir *et al.*, 2025 found 49% in 110 journals in 2024). Compared to many other

fields, ecology and evolutionary biology are at the forefront of data sharing (Tedersoo *et al.*, 2021).

However, despite a high proportion of ecology and evolutionary biology studies archiving data, archived data are often of low quality (Roche *et al.*, 2015, 2022), with most datasets either incomplete (some or all of the data allowing the study to be reproduced is not present) or unusable (e.g., data are not machine readable, in a proprietary format, or are archived with no metadata; Table 1). Based on 362 open datasets from 2013-2019, Roche *et al.* (2022) calculated that 56.4% of datasets were complete, and 45.9% were reusable (out of 362), a situation that has only marginally improved over the last decade (from a sample of 100 articles in 2012/13, 44% were complete, and 36% reusable; Roche *et al.* 2015), with only reusability having statistically increased from 2013 to 2019 (Roche *et al.* 2022). Several studies have further sought to directly assess analytical reproducibility (defined as reproducing the published results using the same data). However, these assessments rely heavily on data provided by authors upon request (Archmiller *et al.* 2020; Minocher *et al.* 2021), as rates of archived data recovery were low (11% in Minocher *et al.* 2021). Conditional on having the full dataset, reproducibility was moderate (42% and 58% of articles were fully reproducible in Archmiller *et al.* 2020 and Minocher *et al.* 2021, respectively), but whether the quality of data provided directly from authors for these studies differs from data that has been archived is not clear. This is similar to other fields; in the *Journal of Psychological Science*, only 9 out of 25 articles were reproducible (given the data) without author intervention (2014/2015; Hardwicke *et al.*, 2021) and even when journals have mandated data archiving only 62% (85/136) of datasets were reusable in the journal *Cognition* (2015/2016; Hardwicke *et al.*, 2018). Overall, these results suggest that most studies across fields either do not have archived data or provide a dataset that has limited utility, but when full datasets are provided, reproducibility can be high. The lack of high-quality data impedes all the goals of data archiving (Box 1).

The use of code for data preparation and analysis is now almost ubiquitous, particularly using the R coding language (Lai *et al.*, 2019; R Core Team, 2022). Increasingly, journals encourage or mandate code archiving (15% in 2015 Mislan *et al.*, 2016; 75% in 2020 Culina *et al.* 2020; 88.4% in 2024 Ivimey-Cook *et al.* 2025). However, the actual rates of code archiving still remain low (2015-2016: 2.5%, 2018-2019: 7.0% in journals without a code archiving policy, Sánchez-Tójar *et al.*, 2025; 23% in 2015/16 to 30% in 2018/19 in journals that encourage or mandate, Culina *et al.* 2020; 18% 2018-2020 in 5 major ecology journals, Kimmel *et al.* 2023). Several recent studies have further tried to assess computational reproducibility (the ability to reproduce the results given the archived data and code; Box 1) but have concluded that it is likely to be low in ecology and evolutionary biology. Kambouris *et al.* (2024) found that out of 177 meta-analyses in ecology and evolutionary biology from 2015-17, only 26 provided both data and code. From these, only 7 studies (27%) could be exactly reproduced (with the results of 15 (58%) studies being reproduced to within 10% of the original results). Kellner *et al.*, 2025 found 7% of 497 articles on species distribution and abundance from 2018-2022 had code that ran. Trisovic *et al.*, (2022) found that out of 9,000 unique R files from the Harvard Dataverse, 74% failed to complete, which lowered to 56% when basic cleaning was applied. The lack of high quality code impedes all the goals of transparency and computational reproducibility (Box 1). Together, the lack of functional data *and* code in public repositories limits the verifiability of published empirical research claims (Henderson *et al.*, 2024) and ultimately erodes trust in science.

Alongside several high profile fraud scandals (Box 1), the lack of adherence to journal archiving policies, and the low quality of data and code archiving has led several journals to recruit data editors (https://www.amnat.org/announcements/data-and-code-announcement.html, Thrall *et al.*, 2023; Barrett, 2024; Barrett & Montgomerie, 2025). Data editors are responsible for

screening and assessing the archived data and code of manuscripts being reviewed by a journal and to assist authors in complying with journal mandates on data and code provision - hereafter, we refer to this process as *data and code quality control*. It is worth stressing that data editors are not acting as gatekeepers; the role of a data editor is to help authors adhere to community standards of data and code archiving. At the time of writing, we are aware of seven journals in ecology and evolutionary biology that have data editors that screen the data and code of some or all manuscripts that are published (American Naturalist, Behavioral Ecology, Ecology Letters, Ethology, Journal of Evolutionary Biology, Proceedings of the Royal Society B, and Peer Community Journal). Behavioural Ecology and Sociobiology has an editor with a related role but only screens a small number of manuscripts at the request of other editors, and additionally provides statistical support (Fernández-Juricic *pers. comms.*).

Data and code quality control by data editors is primarily for the benefit of the authors. A large part of the data editors' role is to help authors ensure their data and code more closely adhere to the open principles that we have adopted as a research community. At the end of the process, authors will therefore have higher quality archived data and code for each publication, which has been linked to increased citation rates (Piwowar *et al.*, 2007; Piwowar & Chapman, 2010; Christensen *et al.*, 2019; Maitner *et al.*, 2024) and increases the prospect of future collaboration based on using and developing archived data and code. Having well archived and documented code also provides a clear advantage for Early Career Researchers that may pursue careers outside of academia, where a proven ability to generate reproducible code is often more of a selling point than publications (Allen & Mehler, 2019; König *et al.*, 2025). There are also many benefits to working reproducibly (Markowetz, 2015), from helping with the continuation of research, to avoiding errors that could later influence results and ultimately require correction or retraction of published work. Data and code quality control further forces

authors to be doubly sure that their dataset is accurate and that the code they used generates the expected results.

While authors benefit most, we believe there are also a multitude of additional benefits for journals, readers and the wider research community. For instance, it is in a journal's best interest to be the purveyor of high quality and reliable scientific research. Adopting data and code quality control signals to readers and the general community that scientific quality and transparency are priorities for the journal. By increasing transparency of analyses, data and code quality control can allow a journal to build its reputation as a reliable and trustworthy source of high-quality science. Through the provision of higher quality data and code, quality control increases the impact of both the article being evaluated, and the journals in which the manuscript is published. Furthermore, it allows other authors and researchers to extend and reuse data and code for further analyses, which can lead to an extended impact for both the original journal and author(s). Finally, ensuring the archiving of high-quality data and code facilitates rigorous post-publication evaluation of claimed results. By increasing transparency and reproducibility of analyses, data and code quality control will therefore increase the trust of published work.

We hope that an emphasis on the quality of archived data and code may additionally help to facilitate data and code review (the detailed evaluation of code; Ivimey-Cook *et al.*, 2023) within research groups prior to submission, creating more opportunities to actively involve co-authors in a project and resulting in a robust and healthy lab culture that promotes cohesion. Such practices can inspire early-career researchers involved in the project by promoting open science through practical experiences, while helping to strengthen trust in scientific integrity in the long term.

In this article, we outline what data editors do, discuss whether there are costs to data and code quality control, and then provide detailed guidelines for data editors that can be used for data and code quality control in journals.

Table 1. Glossary of key terms

Term	**Definition**
Metadata	Refers to a description and information about the data and code. Typically, in the form of a text file called a README. Other variations on this are possible, e.g., a codebook.
Data Editor	An editorial position at a journal. The responsibility of this editor is to screen and quality control the data and code that will be publicly archived alongside manuscripts under review at the journal.
Data and Code Quality Control	The process of checking the suitability of data and code for public archiving.
Data and Code Archiving	The process of depositing data and code in a public repository.
FAIR principles	Findable, Accessible, Interoperable, and Reusable principles for Data (Wilkinson *et al.* 2016) and code (Barker *et al.*, 2022). See https://www.go-fair.org/fair-principles/
Raw data	Unprocessed and unfiltered data. This would include any raw files e.g., photos, audio recordings, videos, and data sheets.
Data Filtering	The process of removing some data to create the dataset used in the analysis (e.g., removal of individuals with a certain characteristic). We refer to the resulting data as *Filtered Data.*
Data Processing	Transforming data from one form to another. Includes data that is extracted from images or videos, data that has been summarised,

	transformed, or is the result of calculations. We refer to the resulting data as *Processed Data.*

What is Data and Code Quality Control and What is it Not?

Data and code quality control by data editors is about increasing the quality of the archived data and code, and ensuring that they meet minimum standards (e.g., the data are complete and usable, and the code is documented and runnable). The guidelines we lay out in the section below give a detailed explanation of what data and code quality control by data editors entails. Ideally, data editors would ensure that archived data and code achieve the goals laid out in Box 1, to allow data reuse, to increase transparency, and to provide computational reproducibility. The importance of these goals may vary across different groups; journals will likely focus more on transparency, whereas readers may be more concerned with data reuse and computational reproducibility. These goals require varying levels of code and data checking, and so, during the early stages of journals recruiting data editors, not all of these goals may be achievable. We also acknowledge that not everyone may agree that achieving all of these goals is the ultimate objective of data and code quality control by data editors. While the importance of each goal may differ among stakeholders, they each help improve the openness, reliability, and transparency of the scientific publication process.

Importantly, whilst data editors are responsible for checking that the archived data and code meet certain minimum standards, they are not responsible for *reviewing* data and code, and so data editors will rarely themselves detect errors or fraud. Data quality control is not about verifying the actual data (e.g., detecting data fabrication) but rather ensuring that data is available, in the appropriate format, and has the corresponding metadata to be scrutinised. The presence of a data editor at a journal will therefore not necessarily prevent fabricated data being

published. We also make a clear distinction here between code *quality control* and code *review* (Ivimey-Cook *et al.*, 2023; Hillemann *et al.*, 2025). Code review is an important part of research (Ivimey-Cook *et al.*, 2023) and we encourage that research groups engage in this practice as a way to improve the quality of published research (Bavota & Russo, 2015). However, data editors are not experts in every field of study, nor are they statisticians or specialists in all programming languages. Therefore, data and code quality control should not extend to assessing the suitability of analyses or code itself.

Are there Costs to Data and Code Quality Control?

Although journals adopting data and code quality control will increase the quality of data and code archiving associated with published articles, which we believe will have widespread benefits to journals, authors, readers and the wider research community (see above), we acknowledge there may be some costs to the widespread adoption of this process.

First, adopting data and code quality control may present an additional burden for a journal. This is likely to primarily impact the length of time required for peer review. To mitigate this problem, several journals currently have data and code quality control alongside peer review (see *Suggestions for Journals* at the end of this article). For journals that currently have in-house data editors, data and code quality control does not create a per-manuscript burden to find extra reviewers.

The process of data and code quality control may add a time burden to authors (although not if authors were already adhering to many journals' existing requirements on data and code archiving). However, this time burden will reduce over time as data and code quality control becomes standard practice and making well-documented data and code becomes a natural part

of a researcher's workflow. This short-term investment will also come with both short and long-term benefits as outlined above. Increasingly, data and code management skills have wide applicability and are becoming part of routine teaching at undergraduate or postgraduate level (Kohrs *et al.*, 2023). We acknowledge that the costs to authors will fall disproportionately on those with less access to training on open data and code practices. However, these researchers are actually those that may benefit most from the process of data and code quality control, which is designed to aid researchers adhere to data and code archiving requirements. Those with lower access to training therefore stand to gain the most from interactions with data editors and the resulting skills learned from increasing the quality of their data and code archiving.

Finally, ensuring that all data and code are archived and checked for computational reproducibility will necessitate more resources for data storage and re-running potentially computationally expensive analyses. Data storage is already reported as a leading cause of increased carbon footprint (https://direct.mit.edu/imag/article/doi/10.1162/imag_a_00043/118246/Ten-recommendations-for-reducing-the-carbon). However, we would argue that to prevent data loss, the data and code behind any study should always be responsibly archived, regardless of the process of data and code quality control. As we outline in the guidelines below, we also do not advocate for the storage of multiple instances of the data, if it is already stored in a public archive. Re-running analyses will also have an environmental impact. However, there is a limit to what data editors are reasonably expected to re-run, and so it is unlikely that highly computationally expensive analyses will routinely be repeated.

SORTEE Guidelines for Data and Code Quality Control

At the time of writing, the practice of data and code quality control is highly variable both across and within journals. To address this, the Society for Open, Reliable and Transparent Ecology and Evolutionary biology (SORTEE) started a working group with 22 data editors from 6 journals in Ecology and Evolutionary (EE) biology, comprising American Naturalist, Behavioural Ecology and Sociobiology, Ecology Letters, Journal of Evolutionary Biology, Peer Community Journal, and Proceedings of the Royal Society B. The goal of this group was to propose a set of structured guidelines for standardising the process of data and code quality control across all EE journals. As a whole, these guidelines provide a high bar for data and code quality control, and so either all or parts of these guidelines can be adopted by journals and presented to authors and readers. We note that validation of Stages 1-4 of these guidelines *by data editors* achieves the highest level (Level 3) of the Transparency and Openness Promotion Guidelines (TOP2025; Grant *et al.*, 2025), for data and code.

We see several benefits of adopting a set of standardised guidelines both for authors and data editors. First, authors know what is expected of them regardless of the journal. By working towards the same standards of data and code archiving in advance of submission, authors can easily submit their manuscripts to a variety of journals, or transfer their manuscripts between them, and know that they do not need to change their submissions to meet different standards across journals. Ultimately, this reduces the burden for authors and streamlines the submission process. In addition, standardisation of this workflow will encourage authors to share data and code even when not explicitly required, help those with less experience of sharing data and code, and facilitate the community-wide adoption of open data and code as a default. Second, data editors have a standardised template for review, designed by both open science advocates and active and experienced data editors from multiple journals, to gain the balance of idealism

and practicality. This can make the process both more efficient for data editors and more thorough for the community. Ultimately, standardisation will help inform decision-making for the journals. Third, readers know what checks have occurred prior to publication, which can ultimately help build additional trust in scientific reporting and open science practices, particularly when computational reproducibility has been assessed.

In an ideal scenario, data and code quality control helps achieve the goals of data reuse, transparency, and computational reproducibility (Box 1). In reality, depending on the time, computing resources, and expertise of the reviewing data editor, only some of these will be feasible at a large scale. We propose that data and code quality control can be broken into six stages in the following coherent order to sequentially address the goals of data and code archiving outlined above (Figure 1):

- Stage 1: Data must be archived and adhere to FAIR guiding principles.
- Stage 2: Archived data corresponds with the data reported in the manuscript.
- Stage 3: Code must be archived and adhere to FAIR guiding principles.
- Stage 4: Archived code corresponds with the workflow reported in the manuscript.
- Stage 5: Archived code runs with the archived data.
- Stage 6: Results can be computationally reproduced by running the archived code.

We discuss each of these stages in more detail below and provide suggestions for how these can be assessed.

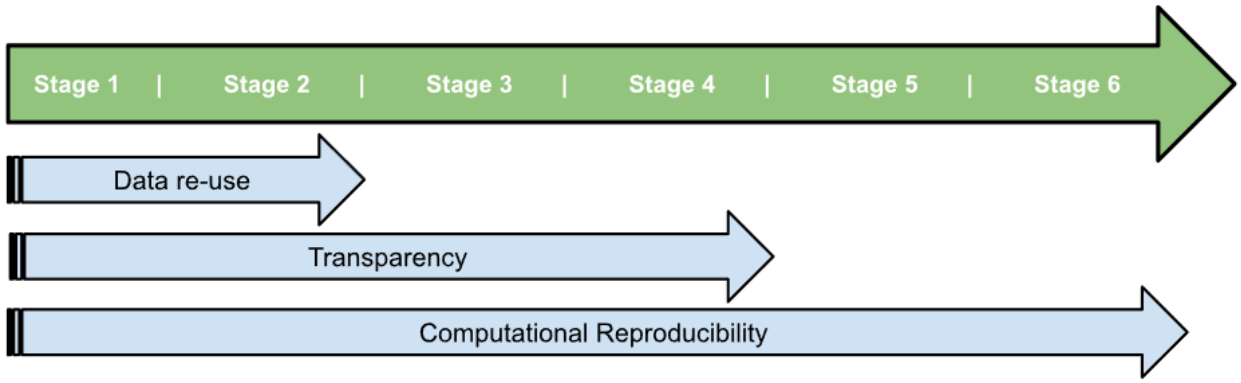

Figure 1. *A diagrammatic representation of how the three goals of data and code archiving (blue arrows) match the stages of data and code quality control (green arrow). Stages 1 and 2 (1 = Data must be archived and adhere to FAIR guiding principles and 2 = Archived data corresponds with the data reported in the manuscript) are needed to achieve the goal of data reuse. Stages 1-4 (3 = Code must be archived and adhere to FAIR guiding principles. and 4 = Archived code corresponds with the workflow presented in the manuscript) are needed to achieve the goal of transparency. Stages 1-6 are needed to achieve the goal of full computational reproducibility (5 = Archived code runs with the archived data and 6 = Results can be computationally reproduced by running the archived code). Figure created by EIC.*

Stage 1. Data must be archived and adhere to FAIR guiding principles.

For data to be open and amenable for reuse, it must adhere to FAIR guiding principles (Findable, Accessible, Interoperable and Reusable). Data must be placed in an open repository (*Accessible*) with a permanent Digital Object Identifier (DOI, or another globally unique and persistent identifier) that is cited in the manuscript (*Findable*) with a licence that describes reuse (*Reusable*). Metadata (e.g., a README; Table 1) must also be present to describe the data. The data itself must be in a machine-readable, non-proprietary format (*Interoperable*). We discuss each of these points in more detail below. By a data editor assessing the archived data to Stage 2 of these guidelines, journals would achieve TOP 2025 level 3 for Data Transparency (Grant *et al.* 2025).

Stage 1.1. Data files are accessible and in an open repository

For data to be readily Accessible and Findable, it must be archived in a public data repository with an associated persistent DOI (or any other globally unique and persistent identifier, such as ARK (for archives, and datasets), Handle (for digital objects), Accession Numbers ('omics' data, e.g., GenBank), RRID (research resources)) that is separate from the DOI of the resulting published article. Furthermore, the data must be clearly cited in the manuscript and listed with its DOI (or other identifier) in the reference list, so readers know where to access the underlying data. There are a multitude of different repositories to fit a variety of needs (see https://zenodo.org/records/10651775 for information about repositories). An important feature of a repository is that it guarantees long-term storage and file immutability (i.e. they cannot be deleted or modified once published). Common general repositories that provide these include DataVerse, Dryad, Figshare, and Zenodo. There are also additional topic specific repositories, such as GenBank for depositing genetic and other biological sequences. We note that whilst GitHub is popular, it does not produce DOIs and files can be changed or deleted, and so projects that use GitHub should be linked to a data repository for archiving prior to submission (e.g., Zenodo). Similarly, the Open Science Framework (OSF) does not provide file immutability; projects and files can be changed or deleted. Data should not be provided as supplementary material attached to the manuscript online as this does not provide a globally unique persistent identifier and may not be open and accessible; all data must be archived in a public repository (see Stage 1.3. for sensitive data).

When necessary, public data repositories can be anonymised to adhere to the journal's policy of double blinding (see: https://methodsblog.com/2023/08/23/double-anonymous-peer-review-frequently-asked-questions/). Furthermore, many repositories offer embargoes, if necessary.

Guideline 1.1.: Data are open and freely available (with some exceptions, see Stage 1.3.) and are located in a permanent public repository with an associated globally unique persistent identifier, that is cited in the text and reference list of the manuscript.

Stage 1.2. Data are associated with a license

Data must be associated with an appropriate license that indicates how the data can be shared and reused. In our experience, most researchers have little knowledge of such licenses. Without a license, data cannot be legally reused under many circumstances (e.g., depending on the jurisdiction; https://choosealicense.com/no-permission/). Therefore, to avoid confusion, it is important for authors to specify a license that outlines how their data can be used, and whether attribution is required. There are several different licenses to choose from but typically Creative Commons licenses are used for data (see: https://chooser-beta.creativecommons.org/), with several repositories including a license by default. The most permissive license is the CC0 license, which puts the data freely into the public domain with no requirement for attribution. Some repositories assign this license by default (e.g., Figshare) or mandate the use of this license (e.g., Dryad). Another commonly used license is CC-BY 4.0, where reusers of the data must give credit to the original author but are allowed to distribute, remix, adapt, and build upon the created material (including for commercial uses). This license is also used as a default by some repositories (e.g., Zenodo). There are several other more restrictive licenses, for example the CC-BY-NC 4.0 which prohibits commercial use.

Guideline 1.2.: Data must be associated with a license.

Stage 1.3. Data files are present and complete

The simplest but most important requirement is that the data supporting the results presented in a manuscript must be complete in the repository. Ideally, raw data and processed data should

393 both be provided (Table 1). The term “raw data” refers to all collected data prior to any filtering
394 (subsetting of data based on reported exclusion criteria) or processing (extraction,
395 transformation, summarising, aggregation, and prior to any formal calculations; Table 1). Note
396 that we do not count transcribing data from a written to a digital format and subsequent error-
397 checking of data entry errors as processing or filtering.
398
399 There are several reasons why the raw data should be archived: First, to prevent data loss,
400 which is achieved by archiving the most complete dataset possible; Second, to maximize data
401 reuse, as only providing filtered data can exclude particular future uses; Third, the process of
402 filtering and processing data is prone to mistakes (e.g., coding errors). Such errors are a natural
403 and inevitable part of the research process, but being able to detect them makes the scientific
404 process more efficient and reliable, and identifying and correcting these mistakes is only
405 possible if the raw data are available. Finally, to increase transparency, allowing the reader a
406 clearer insight into the process that resulted in the final dataset used for analysis. We note that
407 the data required to be archived also depends on the goal; computational reproducibility of the
408 results presented in a manuscript can be achieved with processed data, whereas the goal of
409 data reuse is dependent on raw data being archived. What constitutes raw data is often reliant
410 on the nature of the data (see below). Ultimately, whether the archived data are most
411 appropriate for a given manuscript will be at the discretion of the data editor and dependent on
412 journal policy. Below we provide some guidance for specific cases.
413
414 In the simplest case, data have been collected for a stand-alone study. In this case, the raw
415 data are simply all the collected data, and should be provided in full (for exceptions see below).
416 If data originates from videos, images or sound files, then these are considered to be raw data.
417 Therefore, where possible, these files should also be made available. The processed data
418 should be provided alongside the raw data, with a description of the processing or filtering in the

metadata (if this is not already described in the code files). This is particularly important if the raw data are not interoperable (e.g., outputs from proprietary software; see 1.4 below). Although most databases allow a considerable amount of data to be stored (Per project: Dryad - 300 GB, Zenodo - 50 GB, Figshare - 20 GB, Dataverse - 10GB), raw data may exceed these limits (e.g., video data that is several terabytes large). In such cases where the data are too large to be feasibly uploaded, then a sample of this raw data should be provided, so the extraction process can be assessed (e.g., providing 10 example videos).

If the data used in an analysis originates from a larger database (e.g., from a long-term study) then ideally the entire database would be considered the raw data. We can foresee many circumstances where the authors may feel this is inappropriate, for example, due to worries about the data being used without permission (Mills *et al.*, 2015; but see Evans, 2016 for an empirical assessment) or misused (Weissgerber *et al.*, 2024). In such cases, filtered data may be provided, alongside clear details of the filtering process that would allow the same data to be extracted at a later point (e.g., location and version of the database that the data was extracted from, how it can be accessed, the database queries used to extract the data or other similar instructions of how to generate the same subset for analysis, and any exclusion criteria). Data editors may need to assess the suitability of archived data on a case-by-case basis to ensure that the data are provided in the rawest form possible according to the journal guidelines, and that sufficient information on the generation of archived datasets is present. If the database is already open, rather than re-archiving the data (which, if large, may come with environmental costs) the authors can cite the database, include a clear description of what data were used, the data extraction procedure, and where appropriate, provide an immutable snapshot of the database if it is dynamic.

In some cases, restrictions may apply to making raw data publicly available. For instance, if the dataset contains sensitive information about geographic locations pertaining to endangered species or fossil sites at risk of vandalism (Chapman, 2020). In many cases, data can often be obfuscated or anonymised to enable data archiving. There may also be issues with indigenous data sovereignty (for best practices on governance and stewardship of indigenous data in combination with FAIR principles see CARE principles (Collective benefit, Authority to control, Responsibility, Ethics); Carroll *et al.*, 2021). Processed data used in the manuscript should instead be provided alongside suitable metadata which describes the raw data in as much detail as possible, while still preserving anonymity and sovereignty. Where data cannot be provided, simulated data with the same structure and properties could also be provided, to allow for Stage 5 to be assessed. Information about how and where to make data requests should also be included in the metadata.

In all cases, the data availability statement in the manuscript should clearly outline whether the authors have archived raw and/or processed data. This section should also contain guidance on how to access and request the raw data if necessary and appropriate.

<u>Guideline 1.3.:</u> Authors must either provide:

a) raw data, along with the processed data and/or code to prepare the data for analysis, or b) a sample of raw data alongside processed or filtered data when full raw data upload is not possible, or c) processed or filtered data with a detailed description of how to both obtain and process or filter the raw data.

Stage 1.4. Data files are in an interoperable format

To both facilitate review and allow reuse, data must be in a universally interoperable format, meaning that the data can be exchanged and used across different software and operating

systems. File types specific to proprietary software (e.g., .sps files from the SPSS program) are not interoperable, so do not facilitate data reuse. For example, .xls files are a proprietary format, whereas .xlsx files are not, meaning they are interoperable. However, .xlsx files can contain information that is lost when importing data into statistical software (e.g., formatting). Similarly, tabular data are sometimes archived in a. RData file (or equivalent). Although this can be used with open source software (i.e. R), again, this data format restricts its use, as it requires knowledge of R to extract the data, and may be dependent on the version of R that was used to save it. Simpler text-based file formats such as .csv (comma-separated-values),. tsv (tab-separated-values) and .txt (plain text) files provide a more interoperable format, as they can be used by more software and across more systems, and so are preferable. Where possible, it would therefore be more suitable to archive the raw data in a more interoperable format (i.e., .csv or .txt). Lastly, data should not be stored within PDF or Word documents, which can be prone to error when data are copy-pasted (e.g., for re-using) and which cannot be readily imported into statistical programs for analysis. In some cases, there might be no option other than to provide data in a non-interoperable format (e.g., if the data was collected using proprietary software) but this should be provided alongside extracted data with a clear description of the conversion processes in the metadata including the particular software version that was used.

Guideline 1.4.: Data files must be provided in an interoperable format.

Stage 1.5. Data metadata present and adequate

Data files alone do not contain enough information for a user to fully understand their contents. Data files must therefore be accompanied by metadata (Table 1). The most common form of this metadata is a README file, which describes and explains the content of the data and its provenance. The README should provide general information about the manuscript, e.g., the

manuscript title and abstract, the authors and relevant contact information, date and location of data collection, and a list of all relevant funders. In the case of double-blind review, some sections can be left blank until acceptance (see example in Figure 2). The README should also include any relevant licence information (e.g., CC-BY, see above), and information about data derived from other sources (e.g., from other articles or online data). Finally, the metadata should contain detailed descriptions of each data file, describing its structure and what variables it contains, what units of measurement they are in, and how they link to the data described in the manuscript, e.g., each column in a .csv should be explained and described (see example in Figure 2). This information can be provided in several ways: 1) as part of the main README file, 2) by creating additional README files to describe the data and code files (as shown in Figure 2), or 3) by providing a codebook for each data file (e.g., a .csv file with a column for column names, and another for the description of the variable). We use the term “adequate” here to describe data-associated metadata that is sufficiently detailed so that anyone can understand the data without needing to read the resulting manuscript to understand its contents.

Guideline 1.5.: Detailed metadata, including (but not limited to) a README file, must accompany the data (see example in Figure 2).

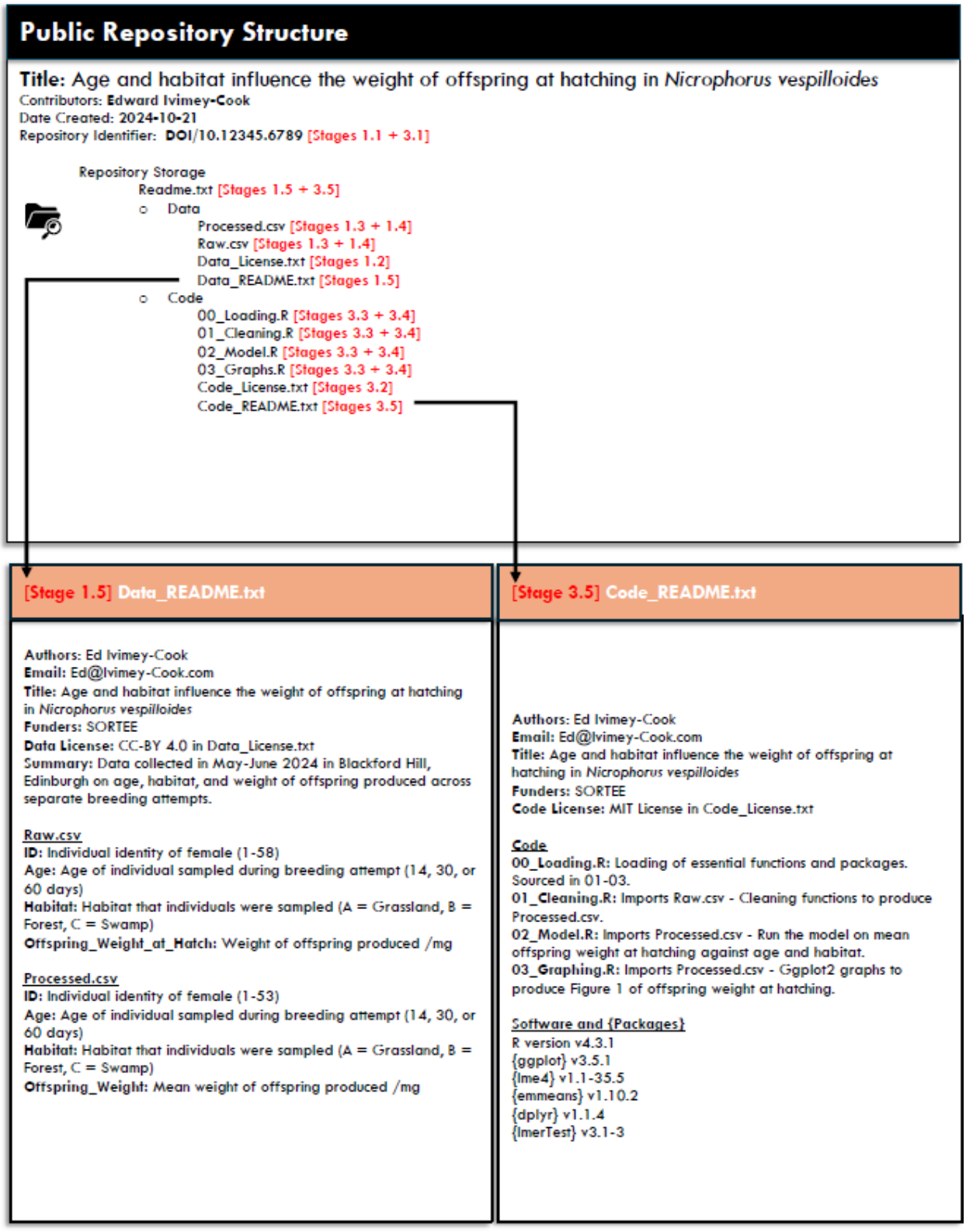

Figure 2. *Example repository structure and metadata (two README files) showing how the various components adhere to the SORTEE guidelines for data and code curation. The numbers in red refer to*

the stages being addressed. The Data README should contain information on the manuscript (authors with corresponding contact details, the title of the manuscript, and any funders), the license file, along with information of the data (a brief summary of collection, and column-by-column description of the data files along with any measurement units or levels of factors). For code, the README should contain the same information as the data initially (information on the manuscript and code license) but also contain a description of each code file in the order they are meant to be used (which also clearly indicates which data file is used in each script). Lastly, the README should contain a list of all software and packages used with associated version numbers. Figure created by EIC.

Stage 2. Archived data corresponds with the data reported in the manuscript

For archived data to support a manuscript, as well as being present in a form that facilitates reuse, it must correspond with the data reported in the manuscript. For this to be assessed, the data editor needs to check that the variables and data described in the manuscript (most likely in the Methods) are present in the data files provided. For example, if the manuscript mentions that offspring weight was measured at three habitats, the data file should contain an offspring weight variable and a habitat variable (see example in Figure 3). The dimensions of the data should also correspond with those described in the manuscript; discrepancies in the size of the dataframe may suggest that some unreported data processing or filtering has taken place. A clear description of all these aspects within the text is essential; without it, the data will not correspond with the manuscript, undermining its potential for reuse, transparency and reproducibility. Some journals use AI to facilitate this process (e.g., the DataSeer.ai application: https://dataseer.ai/), which produces a report detailing the expected data that should be provided based on the description within the manuscript.

Guideline 2.: The structure and contents of the archived data files must match the description in the manuscript.

Manuscript

Methods

We monitored a population of Nicrophorus vespilloides over three habitats (Grassland, Forest, and Swamp) on Blackford Hill, Edinburgh from May-June 2024. Over this time, we discovered 58 broods (23 Grass, 20 Forest and 15 Swamp), from which we took one larva and brought them back to the lab for rearing. We then monitored three breeding attempts (at 14, 30, and 60 days post-eclosion) for each of the resulting beetles. For each breeding attempt, we measured the hatching weight of offspring born to each brood (in mg). Five individuals died during the experiment and so we removed them from the dataset before analysis. In total we analysed records of 159 broods (53 females each with three broods). To test whether age and habitat affected hatching weight, we first took the mean weight of offspring at hatching per individual per age class. We then modelled hatching weight as a function or age (continuous) and habitat (factor), including a random factor of individual ID in a linear mixed model. [Stages 2 + 4]

Results

The effect of age on offspring weight was significantly negative (Age = -0.03, SE = 0.004, $p<0.001$). [Stages 5 + 6]

[Stages 2 + 4] Processed.csv

	ID	Habitat	Age_days	Offspring_Weight_g
2	1	A	14	0.22
3	1	A	30	0.33
4	1	A	60	0.21
5	2	B	14	0.28
6	2	B	30	0.24
7	2	B	60	0.22
...	...	...	...	...
160	53	C	60	0.34

[Stages 2 + 4] 02_Model.R

```
####02.1 Modelling code

#Load packages
source("00_Loading.R")

#Import processed data
Processed_data <-
read_csv("Processed.csv")

#Lmer of offspring weight with age and
habitat #with a random effect individual
ID - requires #"Processed.csv"
model1 <- lmer(Offspring_weight_g ~
                Age_days +
                Habitat +
                (1|ID),
          data = Processed_data)


summary(model1)
```

[Stages 5 + 6] Terminal

```
> model1 <- lmer(Offspring_weight_g ~ Age_days + Habitat + (1|ID), data = Processed_data)
> summary(model1)
Linear mixed model fit by REML. t-tests use Satterthwaite's method ['lmerModLmerTest']
Formula: Offspring_weight_g ~ Age_days + Habitat + (1 | ID)
   Data: Processed_data

Fixed effects:
             Estimate   Std. Error         df   t value  Pr(>|t|)
(Intercept)   5.051133   0.196697   195.718112   25.680    < 2e-16 ***
Age_days     -0.030746   0.003729   191.946582  -8.246    2.54e-14 ***
HabitatB     -0.056885   0.090364   189.647801  -0.630    0.530
HabitatC      0.058789   0.091050   188.100452   0.646    0.519
```

Figure 3. *Matching a manuscript to archived data and code. The methods in the manuscript can be checked against the archived data (Stage 2) and code (Stage 4). In terms of the data, the same variables that are described in the manuscript need to be present in the data, and the data need to be the same size as referred to in the manuscript. In terms of the code, the models that are described in the*

manuscript need to be clearly labelled in the code. The numbers in red refer to the stages of the guidelines being addressed. Figure created by EIC.

Stage 3. Code must be archived and adhere to FAIR guiding principles

To facilitate transparency and computational reproducibility, all code used to reproduce the results should be provided alongside data. The guidelines for code archiving are broadly similar to those of data archiving outlined above, with a few subtle differences, which we outline below. By a data editor validating Stages 3 and 4 of these guidelines, journals would achieve TOP 2025 level 3 for Analytic Code Transparency (Grant *et al.* 2025).

Stage 3.1. Code files are accessible and in an open repository

As with data, code files must be accessible within an open repository with an associated globally unique persistent identifier (see Stage 1.1 for suitable repositories). This can either be the same repository as the data or a separate one. This choice may depend on the chosen repository. For example, Dryad recommends only archiving data, and directs users to archive code with Zenodo, as code is not always compatible with a CC0 license, which Dryad mandates. Care must be taken when archiving code and data separately, as the two repositories should clearly link to each other as well as to the manuscript, and authors must provide information about how to structure the data and code directories so that the code will run with the data. For example, if the code assumes that the data are in the same parent directory within a folder called ‘Data’ (see Figure 2), then the data files will have to be organised like this for the code to run. Where possible, we suggest that data and code are archived together as this will minimise issues with computational reproducibility (see Stage 5). Similar to data, code should not be included in the supplementary material of a submission. Again, if necessary, the repositories can be anonymised (see stage 1.1 above).

Guideline 3.1.: Code files are open and freely available and are located in a permanent public repository with an associated globally unique persistent identifier, preferably in the same repository as the data files, and the repository is cited in the text and the reference list of the manuscript.

Stage 3.2. Code is associated with a licence

As with data, any archived code must also have an associated license to enable code sharing and repurposing. It is worth noting that typically licenses used for code differ from those used with data (i.e. Creative Commons licenses), and there is a multitude to choose from. We suggest using permissive code licences whenever possible, for example, MIT, BSD 2-Clause, GNU, and Apache (see: https://choosealicense.com/).

Guideline 3.2.: The code must be associated with a license.

Stage 3.3. Code files are present and complete

Without code alongside the data used in the analysis (and the computational environment in which the analysis took place – see Stage 3.5), full transparency and computational reproducibility are impossible to achieve. At a minimum, the analytical code used for statistical analyses and graphing should be present, but we recommend providing all parts of the analysis pipeline, from data filtering and processing to model analysis and graphing.

Analyses are not always done in programming languages (e.g., R). However, several analytical programs (particularly those that use Graphical User Interfaces (GUIs)) will output a script or log detailing the analysis procedure (e.g., SPSS or Minitab), which should then be archived. If this is not possible, the researcher should clearly document which menu options were selected in the GUI and in what order, with sufficient detail to enable reproducibility. Alternatively, they can

provide screenshots showing all selected options during the analysis. It should be noted that these output scripts or instructions with GUI-based software are often proprietary so will still limit reproducibility (discussed further in Stage 6).

Guideline 3.3.: All code used for generating the results of the manuscript (including filtering, processing, graphing, and analysis) must be present either in one or more code files.

Stage 3.4. Code is in an interoperable format

For code files to be opened and usable, it is essential they are provided in an interoperable format such as a text (.txt), R (.r) or Python (.py) file. These file formats can be readily opened by text editors and other integrated development environments (e.g., VSCode). Code must not be provided as a PDF or pasted into a Word document, because even if the script can be copied and pasted, this increases the risk of unintentional errors, as these programs often insert additional characters (or spacing) that can be misinterpreted by the analysis software (e.g., Python code failing to run due to improper indentation).

Guideline 3.4.: Code files must be provided in an interoperable format.

Stage 3.5. Code metadata present and adequate

Code metadata must be present in two different forms, in a separate README file and also within the code file itself. As with data, a detailed README file must be provided along with the code files that describes general information about the manuscript, e.g., manuscript title and abstract, the authors and contact information, list of all relevant funders, the globally unique persistent identifier of related data (if different), and information about the code license. Additionally, the README file should also include an outline of the workflow of the code (if

multiple files exist), how to use it with the data (if archived separately) and a brief description of each code file including what data they require (i.e. raw or processed data), what they do (i.e. filtering, processing, modelling etc) and what they produce (e.g., Figure X or Table X). Finally, the README should include details of the name and version of the analytical software used (e.g., R or Python) along with the names and version numbers of the loaded (not base) packages used (e.g., these can be obtained using sessionInfo() in R). This information is essential for detailing the computational environment and enabling computational reproducibility.

The second form of metadata is included within the code files, in the form of detailed code annotation and sectioning. A header at the top of the script with a title and a quick overview of what the code does can also be very helpful, especially when the whole analysis pipeline is split across multiple scripts. The code should be broken down into distinct sections with clear headings describing their purpose (e.g., loading packages, data processing, data filtering). Annotation should then clearly describe what the code does, how to run it (if necessary, the length of time it might take, for example if the code takes multiple hours to complete), and what it produces. From the perspective of a data editor, the most important thing is that sections of the code are clearly signposted to help assess Stage 4 (see below). Therefore, line-by-line annotation, whilst important to readers and users, is not as vital as clear labelling of sections of code and their purpose for data editors. There is no expectation that the data editor will understand all the code they quality control, and it should not be the role of data editors to review, interpret, or correct the code. Similar to the data metadata (Stage 1.5), the term “adequate” refers to the code metadata providing all the information necessary to understand the analysis code without reading the manuscript.

Guideline 3.5.: A sufficiently detailed README must accompany the code. Code must also be broken into sections with clear annotation stating the purpose of the code with clear links to the relevant sections, figures, and tables in the manuscript.

Stage 4. Archived code corresponds with the workflow reported in the manuscript

It is crucial that all the code needed to reproduce the results of the manuscript, including any supplementary material, is archived (Figure 3). This stage should not involve the data editor critiquing the analytical techniques used or performing a formal code review (Ivimey-Cook *et al.* 2023, Hillemann *et al.* 2025; although annotation is required for transparency, see Stage 3.5). It should rather involve an assessment of whether the specific code is present to perform all stages of the analyses, including producing any graphs and subsequent results stated within the manuscript. At this stage, we are also not interested in whether the code reproduces the results in the manuscript, just that the code is *present* to produce the results. As a data editor is unlikely to be an expert in all analyses, across all packages or all programming languages, clear code annotation and signposting by the author is necessary for this to be assessed. To our knowledge, there is currently no software that performs the same task for code as DataSeer does for data. However, given the rapid progress in AI, such a tool may become available soon (Cooper *et al.*, 2024).

Guideline 4.: The structure and content of the archived code must match the description of data filtering, processing, and analysis, and the presentation of results in the manuscript.

Stage 5. Archived code runs with the archived data

This stage is a prerequisite for full computational reproducibility. The data editor must be able to run the code with the provided data and code metadata, using the described software, without

errors. The metadata provided must therefore be sufficient for a reader to install appropriate programs and libraries (and their versions) required to run the code, and to understand which code files should be run and in what order. If the data editor cannot run the code with the archived data and metadata, they cannot progress to the last stage of the guidelines, and so the data editor should then ask the authors to fix the issue. Common issues include a package or module not being installed or loaded within the code, a missing code chunk, variable names in the code and data files not matching, and code referencing data files with names that do not match the archived data files. We stress that it is not the responsibility of the data editor to solve these problems and make the code run as intended, but rather the onus should be placed on the authors.

One of the most common reasons code does not run is due to the use of local or absolute file paths that do not transfer to another user's operating system. A more reproducible way of specifying file paths is to use relative file paths and there are multiple ways to do this. A common way for RStudio users is creating an RStudio project file (.Rproj), or similarly using the {here} R package (Müller & Bryan, 2020) outside of RStudio (see https://docs.posit.co/ide/user/ide/get-started/). Alternatively, local file paths can be specified when R is opened, for example by opening R within a certain directory when using the terminal or using an integrated development environment (IDE) that allows users to specify a project folder (e.g., the R GUI or VSCode). Whichever method is used should be noted in the metadata. Given the multitude of methods, the use of absolute file paths or a different method of specifying relative file paths than the data editor is familiar with should not be a reason for a data editor to return the code to an author, as long as the data editor can make it run on their computer with minor changes. We class this as a minor error that should simply be noted in the final review.

Similarly, data editors should not be expected to install exact software versions in the first instance. If there are errors upon running (or results do not match - see Stage 6), then the data editor should note this in their review, and the corresponding version should be installed and code run again. Finding that the results are not reproducible with different versions can be, in itself, an insightful piece of information regarding the robustness of the results.

In some cases, the data preparation or analysis may be computationally expensive and so either require specialist hardware (such as access to a high performance cluster) or take a considerable time to run. This should be clearly indicated within the metadata, alongside a saved output. Ideally, example code should be provided that demonstrates that the code will run. For example, if a statistical model will take a long time to run, the authors can provide example code for an analysis of a subset of the data, or present a model that runs for a reduced duration. Alternatively, data editors could also simply check that the code initiates and then terminate the run before completion. Although this does not allow computational reproducibility to be fully assessed (see Stage 6), it at least demonstrates that the code runs. Similarly, the code may come from proprietary software or use packages from proprietary software (e.g., {ASReml-R}; Butler *et al.*, 2017). In such cases, the data editor will not be able to run the code and so full computational reproducibility cannot be assessed. If this is the case, this should be clearly documented in the metadata. In the case that only part of the analysis requires proprietary software, the metadata should clearly indicate which parts of the code can be assessed by the data editor. As we outline below, in both the case of computationally expensive analyses and the use of proprietary software, where possible the authors must provide saved outputs from these analyses for the data editor to review. For example, outputs of large Bayesian models that can take a considerable time to run can be saved (e.g., as a .RDS file) and archived.

In some cases, the problems of using proprietary software can be overcome by ensuring that the code can be executed using non-proprietary software or by providing alternative executable formats. For example, GNU Octave can be used to run MATLAB code and MATLAB Compiler allows converting MATLAB (.m) files into standalone applications, ensuring data editors and users can run the code without owning the proprietary software. Although such alternatives can provide computational reproducibility, authors must carefully test for compatibility and note any limitations or differences in the metadata.

Guideline 5.: Code must be able to run without error using the archived data. With the exception of easy to fix file path errors, all errors should be addressed by the author.

Stage 6. Results can be computationally reproduced by running the archived code

For this final stage, the data editor should assess computational reproducibility by checking whether the results in text, tables, and graphs within the manuscript and supplementary material match those obtained by running the archived code with the archived data. This can only be assessed if the archived code runs without error (Stage 5). In most cases, we expect that exact reproducibility of the results is possible (i.e., the exact number in the manuscript should be generated by running the code), and any deviations would mean that the computational reproducibility test has failed. In some cases, authors may have used additional software to post-process figures. In these cases, the data represented within the figure is still expected to be the same, but the code may not reproduce the figure exactly.

One reason that the reproduced results may slightly differ is due to the use of stochastic methods that involve (pseudo) random number generation, such as Monte Carlo methods (e.g., simulations or Bayesian analysis using Monte Carlo Markov Chains (MCMC)) as these will produce a slightly different result each time they run. However, this variation can be avoided by

755 setting a seed (e.g., using set.seed() function in R or random.seed() in Python; see Box 1) at the
756 beginning of any code section that would be run independently, which means that the
757 pseudorandom number generation is the same each time the code is run, enabling the same
758 results to be reproduced, including for analyses and figure generation (e.g., with point jittering).
759 We note that setting seed does not always ensure computational reproducibility, for instance the
760 use of rmvnorm() from the {MASS} R package does not create the same random numbers
761 across different operating systems due to floating point errors. If there is no way for the data
762 editor to generate the exact result (e.g., because the software does not allow setting a seed)
763 then the data editor can allow a degree of tolerance for the result which should be noted in their
764 review. Archmiller *et al.* (2020) suggest comparing the conclusion (the direction and significance
765 of results) as well as the numbers of the original and reproduced results. In the first case, if the
766 direction of the effect changes, or the statistical significance changes, then this should be
767 viewed as failing the computational reproducibility test. For results close to zero or the
768 significance threshold, small changes in the results might change direction or significance,
769 respectively. Hardwicke *et al.*, 2021 therefore suggested using % error (i.e. (reproduced–
770 original)/original × 100), as this is not dependent on the scale of the results, where 0-10% was
771 classified as a minor deviation and >10% as a major deviation, and therefore, not reproducible.
772 However, this % error method (1) still allows for a substantial deviation from the reported values,
773 (2) would result in different tolerances for different effects within the same model, and (3) is
774 most meaningful when effect sizes are on a ratio scale, which typically they are not. Perhaps
775 most importantly, reproduced results should fall well within the reported uncertainty of the
776 original result, and if they do not, this should be viewed as a failure to reproduce the results. The
777 data editor should communicate the conditions under which computational reproducibility was
778 assessed (e.g., the tolerance threshold) in their review. As opposed to in-text results and tables,
779 figures cannot be exactly compared without the use of specialist software but should be
780 compared by eye for reproducibility.

The use of computationally expensive methods or proprietary software may mean that the data editor cannot feasibly re-run the analysis in full (see Stage 5 above). If none of the code can be run by the data editor, for example if it all takes place using proprietary software, then computational reproducibility cannot be assessed (both Stages 5 and 6 would fail). Clearly, this should not prevent the publication of a manuscript containing such analyses in journals where data editors assess Stages 5 and 6 of these guidelines. In this case, we would therefore recommend that it is highlighted in the manuscript that computational reproducibility could not be assessed (e.g., in the data and code availability statement or open research sections). If it is only part of the code that cannot be run by the data editor (e.g., a computationally expensive model), then the output of this code should be provided by the author in the repository and noted in the metadata, so that the output can be compared to the manuscript by the data editor.

Guideline 6.: Results reproduced by the data editor with the archived data and code must match those presented in the manuscript. A tolerance threshold can be given when there is not an exact match, but the authors must state clearly in the code metadata why this mismatch might occur. If saved model outputs are instead provided, this must also be clearly stated in the metadata.

Suggestions to Authors

Data and code quality control is becoming increasingly common across journals in ecology and evolutionary biology. Consequently, authors will have to adhere to certain guidelines for data and code sharing. Although the guidelines presented here are largely aimed at data editors, knowledge of the checks that a data editor is expected to perform will help authors understand what is needed from their data and code prior to submission. We hope that the widespread

adoption of these guidelines will make the process more transparent for authors and also consistent across journals in the event of manuscript resubmission elsewhere. We acknowledge that making data and code readily accessible and reusable adds to the workload of authors (at least initially). We have several suggestions to ease this process:

Adhere to the data and code quality control guidelines from the beginning of the research project

Working to make a repository accessible and reproducible at the end of a project is a lot of work. We would recommend creating a clear directory structure and creating metadata (e.g., a README) at the beginning of the project and updating the metadata as new files are added. Similarly, annotating code as authors produce it, not only with section descriptions but also with information about how they run and what output they produce, is far easier than going back and annotating code at the end of the project. Bearing reproducibility in mind while working on a project also makes it far more likely that someone else will be able to reuse the repository and successfully run the code. This is even more useful if authors plan on collaborating with multiple people during the project's lifetime. Generally, there exists a multitude of benefits to working reproducibly (Markowetz, 2015). We acknowledge that for many authors this may present a steep learning curve, however adherence and knowledge of these guidelines will promote learning and progression over time.

Prepare data and code according to the data and code quality control guidelines before submission to any journal

Inherently linked to the point above, if authors have not prepared data and code according to the data and code quality control guidelines from the start of the project, it is advisable to at least have data and code ready for submission. This will minimise any problems during both the

submission process and the data and code quality control and allow for easy transfer between journals.

Perform a pre-submission code review

It is advisable for authors to send their data and code repository to a colleague or co-author for a code review prior to submission to a journal (Ivimey-Cook *et al.* 2023). This enables checking whether the code in the author's repository runs with the data in the repository structure provided. They can then check whether there is appropriate and adequate metadata, whether data and code match the manuscript, and whether the code reproduces the results in the manuscript. Importantly, co-authors may also be more likely to spot any mistakes in the code as they are familiar with the project and data, and data editors do not check the reliability of code. This could be done within research groups, where the task of code reviewing is shared between members of the team, or as part of a larger 'code club'. Open science organisations, such as SORTEE, have their own code clubs which are open to join. For further advice on setting up code clubs see Ivimey-Cook *et al.* (2023).

Consider presenting code and associated outputs using Markdown or Quarto

Presenting everything in one self-contained document such as a Markdown or Quarto file can be very helpful for data editors and future readers or users (Buckley *et al.*, 2025). It allows for a clear link between the code, the data, and the resulting outputs that may need to be assessed.

Suggestions for Journals

Data and code quality control should start at submission

Currently, in the majority of journals, data and code quality control occurs after (or close to) acceptance. We recommend that data and code are required at submission (see above for

methods to anonymise data and code repositories), and that data editors perform a light check of the data, code, and associated metadata (e.g., Stages 1 and 3) early on (before sending to review). This enables reviewers to both see and review the data and code during peer review (if they choose to), and also engages the authors in the data and code quality control process at an early stage. That way, any problems can be highlighted and addressed early in the process. Computational reproducibility checks (Stages 5 and 6) would ideally be conducted later in the process, at a point where the code (particularly related to statistical analysis) is unlikely to change as a result of further review, in order to avoid a data editor having to perform these checks multiple times.

Ensure Journals have data editors with a mixture of coding expertise

There exists a multitude of different languages in which to write code and analyse data. Although R is one of the most popular in Ecology (from 58-80% of studies in ecology and evolution; Lai *et al.* 2019, Culina *et al.* 2020, Kambouris *et al.* 2024) code is often written in other languages such as Python, MATLAB, SAS, Julia, to name a few. It is therefore important that a journal considers having multiple data editors with varying coding language, data type and area expertise. This means that data editors can be suitably paired to each manuscript.

Have clear guidelines on the journal website

Authors will be more likely to adhere to the guidelines adopted by the journal prior to submission if these are clearly displayed on the website, ideally under ‘Instructions to Authors’ sections. These need to outline what stages of the guidelines the data editors check (e.g., Stages 1-4), what they expect at each stage from the author, and what the authors need to state in their data availability statement. Having data, code, and associated metadata already in a state ready for quality control will reduce much of the work for the data editor. Some journals additionally provide template README files to help authors.

Have clear statements within manuscripts

For readers to know what quality control checks have been performed and to highlight the journal's endeavours to ensure the highest quality research, it should be clearly stated within the data and code availability section what checks have been performed. For instance, “Data and code were checked from Stage 1-4 of the SORTEE guidelines for Data and Code Quality Control”. This statement should also contain information if a check has not been able to be performed, for instance, if the use of proprietary software or sensitive data was involved, impeding computational reproducibility tests.

In psychology and medicine, open science badges have previously been used to indicate manuscripts that adhere to certain open science practices (e.g., open data, open code, open materials, pre-registration) with the ultimate goal of encouraging authors to adopt these practices. Evidence for their effectiveness in increasing data and code sharing is mixed, with early observational studies reporting increases in data sharing after badge implementation (Kidwell *et al.*, 2016), but a subsequent randomized controlled trial finding no such effect in a biomedical journal context (Rowhani-Farid *et al.*, 2020). Note that the journals surveyed in these cases did not have data editors actively checking the data and code archiving. The presence of badges has also been shown to increase the trust of researchers in published articles (Schneider *et al.*, 2022). Journals could choose to use such badges following data and code quality control to indicate that presence of open data (Stage 1-2) and open code (Stage 3) has been verified, and further badges could be developed for computational reproducibility (Stages 5-6).

Have clear definitions and policies of what code and data the journal requires

Ideally, all the data and code used to generate the results should be archived, including both raw and processed data and all the code used to process, filter, model, and graph. However, this is at the discretion of the journal and therefore should be made explicit to the authors prior to submission. We recommend that the form of data and code is clearly described in the data availability statement of the manuscript, for instance, “Processed data and code used in modelling and graphing are archived here…”.

Conclusion

Here we present a standardised set of guidelines for data and code quality control for journals in ecology and evolutionary biology. As it stands, rates of data and code archiving, and importantly the quality of archived data and code, are low. By recruiting data editors, journals can positively impact the state of open data and code, and in doing so increase research transparency and reproducibility. With the SORTEE data and code quality control guidelines, we aim to increase the quality and consistency of data and code quality control across journals that currently have data editors and provide a template for journals wanting to start data and code quality control. We believe that these guidelines will have substantial benefits for journals, for authors, and for the wider scientific community.

Acknowledgments

We thank Bob Montgomerie for extensive discussion and comments on the manuscript. Thanks also to Lars Vilhuber for a useful discussion on the role of data editors in Economics.

Author Contributions

Conceptualisation - JLP and EIC

Writing - Original Draft - JLP and EIC

Writing - Review & Editing - All authors

Visualisation - EIC

Project administration - JLP

Supervision - EIC

Conflict of Interest

SORTEE has been financially supported by Dryad, Figshare, the Center of Open Science (which hosts the Open Science Framework; OSF), Peer Community In, the American Society of Naturalists and the Royal Society, all of which are mentioned in the guidelines. EIC is the 2025 president of SORTEE. EIC, ML, MP, AST are on SORTEE board of directors. JLP, KBN, CJ, SN, and EIC are members of the SORTEE advocacy committee. JLP, BJA, KBN, JAB, BC, DG, CJ, RK, ML, SN, ROD, MP, QP, AST, NvD, and EIC are SORTEE members. BB is a data editor at the American Naturalist. EIC, AST, ROD, NvD, MJG, TD, EF, PDA, and QP are data editors at Ecology Letters. BJA, JAB, DG, DSM and LW are data editors at Proceedings B. SL is the data editor at Journal of Evolutionary Biology. EFJ is the data editor from Behavioural Ecology and Sociobiology. BC, RK, ML, and MP are data editors at PCI.

References

Allen, C. & Mehler, D.M.A. 2019. Open science challenges, benefits and tips in early career and beyond. *PLOS Biol.* **17**: e3000246. Public Library of Science.

Archmiller, A.A., Johnson, A.D., Nolan, J., Edwards, M., Elliott, L.H., Ferguson, J.M., *et al.* 2020. Computational Reproducibility in The Wildlife Society's Flagship Journals. *J. Wildl. Manag.* **84**: 1012–1017.

Barker, M., Chue Hong, N.P., Katz, D.S., Lamprecht, A.-L., Martinez-Ortiz, C., Psomopoulos, F., *et al.* 2022. Introducing the FAIR Principles for research software. *Sci. Data* **9**: 622. Nature Publishing Group.

Barnes, N. 2010. Publish your computer code: it is good enough. *Nature* **467**: 753–753. Nature Publishing Group.

Barrett, L. & Montgomerie, R. 2025. A data editor for behavioral ecology. *Behav. Ecol.* **36**: araf077.

Barrett, S.C.H. 2024. Proceedings B 2024: the year in review. *Proc. R. Soc. B Biol. Sci.* **292**: 20250065.

Bavota, G. & Russo, B. 2015. Four eyes are better than two: On the impact of code reviews on software quality. In: *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pp. 81–90.

Belkhir, K., Smadja, C.M., Antoine, P.-O., Scornavacca, C. & Galtier, N. 2025. An overview of open science in eco-evo research and the publisher effect. EcoEvoRxiv.

Berberi, I. & Roche, D. 2023. Living database of journal data policies in E&E. , doi: 10.17605/OSF.IO/D6SP3. OSF.

Berberi, I. & Roche, D.G. 2022. No evidence that mandatory open data policies increase error correction. *Nat. Ecol. Evol.* **6**: 1630–1633. Nature Publishing Group.

Bolnick, D. & Paull, J. 2016. Retraction: Morphological and dietary differences between individuals are weakly but positively correlated within a population of threespine stickleback. *Evol. Ecol. Res.* **17**: 849.

Buckley, Y.M., Bardgett, R., Gordon, R., Iler, A., Mariotte, P., Ponton, S., *et al.* 2025. Using dynamic documents to mend cracks in the reproducible research pipeline. *J. Ecol.* **113**: 270–274.

Butler, D.G., Cullis, B.R., Gilmour, A.R., Gogel, B.G. & Thompson, R. 2017. ASReml-R Reference Manual. VSN International Ltd, Hemel Hempstead, HP1 1ES, UK.

Carroll, S.R., Herczog, E., Hudson, M., Russell, K. & Stall, S. 2021. Operationalizing the CARE and FAIR Principles for Indigenous data futures. *Sci. Data* **8**: 108. Nature Publishing Group.

Chapman, A.D. 2020. Current Best Practices for Generalizing Sensitive Species Occurrence Data.

Christensen, G., Dafoe, A., Miguel, E., Moore, D.A. & Rose, A.K. 2019. A study of the impact of data sharing on article citations using journal policies as a natural experiment. *PLOS ONE* **14**: e0225883. Public Library of Science.

Cologna, V., Mede, N.G., Berger, S., Besley, J., Brick, C., Joubert, M., *et al.* 2025. Trust in scientists and their role in society across 68 countries. *Nat. Hum. Behav.* **9**: 713–730. Nature Publishing Group.

Cooper, N., Clark, A.T., Lecomte, N., Qiao, H. & Ellison, A.M. 2024. Harnessing large language models for coding, teaching and inclusion to empower research in ecology and evolution. *Methods Ecol. Evol.* **15**: 1757–1763.

Culina, A., Berg, I. van den, Evans, S. & Sánchez-Tójar, A. 2020. Low availability of code in ecology: A call for urgent action. *PLOS Biol.* **18**: e3000763. Public Library of Science.

Evans, S.R. 2016. Gauging the Purported Costs of Public Data Archiving for Long-Term Population Studies. *PLOS Biol.* **14**: e1002432. Public Library of Science.

Feng, X., Qiao, H. & Enquist, B.J. 2020. Doubling demands in programming skills call for ecoinformatics education. *Front. Ecol. Environ.* **18**: 123–124.

Fernández-Juricic, E. 2021. Why sharing data and code during peer review can enhance behavioral ecology research. *Behav. Ecol. Sociobiol.* **75**: 103.

Gihawi, A., Ge, Y., Lu, J., Puiu, D., Xu, A., Cooper, C.S., *et al.* 2023. Major data analysis errors invalidate cancer microbiome findings. *mBio* **14**: e01607-23. American Society for Microbiology.

Goldacre, B., Morton, C.E. & DeVito, N.J. 2019. Why researchers should share their analytic code. *BMJ* **367**: l6365. British Medical Journal Publishing Group.

Gomes, D.G.E., Pottier, P., Crystal-Ornelas, R., Hudgins, E.J., Foroughirad, V., Sánchez-Reyes, L.L., *et al.* 2022. Why don't we share data and code? Perceived barriers and benefits to public archiving practices. *Proc. R. Soc. B Biol. Sci.* **289**: 20221113. Royal Society.

Gould, E., Fraser, H.S., Parker, T.H., Nakagawa, S., Griffith, S.C., Vesk, P.A., *et al.* 2025. Same data, different analysts: variation in effect sizes due to analytical decisions in ecology and evolutionary biology. *BMC Biol.* **23**: 35.

Grant, S., Corker, K., Mellor, D., Stewart, S., Cashin, A., Lagisz, M., *et al.* 2025. TOP 2025: An Update to the Transparency and Openness Promotion Guidelines. OSF.

Hardwicke, T.E., Bohn, M., MacDonald, K., Hembacher, E., Nuijten, M.B., Peloquin, B.N., *et al.* 2021. Analytic reproducibility in articles receiving open data badges at the journal Psychological Science: an observational study. *R. Soc. Open Sci.* **8**: 201494. Royal Society.

Hardwicke, T.E., Mathur, M.B., MacDonald, K., Nilsonne, G., Banks, G.C., Kidwell, M.C., *et al.* 2018. Data availability, reusability, and analytic reproducibility: evaluating the impact of a

mandatory open data policy at the journal *Cognition*. *R. Soc. Open Sci.* **5**: 180448. Royal Society.

Henderson, A.S., Hickson, R.I., Furlong, M., McBryde, E.S. & Meehan, M.T. 2024. Reproducibility of COVID-era infectious disease models. *Epidemics* **46**: 100743.

Hennessy, E.A., Acabchuk, R.L., Arnold, P.A., Dunn, A.G., Foo, Y.Z., Johnson, B.T., *et al.* 2022. Ensuring Prevention Science Research is Synthesis-Ready for Immediate and Lasting Scientific Impact. *Prev. Sci.* **23**: 809–820.

Hillemann, F. [freddy], Burant, J.B., Culina, A. & Vriend, S.J.G. 2025. Code review in practice: A checklist for computational reproducibility and collaborative research in ecology and evolution. *EcoEvoRxiv*. EcoEvoRxiv.

Ivimey-Cook, E.R., Pick, J.L., Bairos-Novak, K.R., Culina, A., Gould, E., Grainger, M., *et al.* 2023. Implementing code review in the scientific workflow: Insights from ecology and evolutionary biology. *J. Evol. Biol.* **36**: 1347–1356.

Ivimey-Cook, E.R., Sánchez-Tójar, A., Berberi, I., Culina, A., Roche, D.G., Almeida, R.A., *et al.* 2025. From Policy to Practice: Progress towards Data- and Code-Sharing in Ecology and Evolution. EcoEvoRxiv.

Janssens, M., Gaillard, S., Haan, J.J. de, Leeuw, W. de, Brooke, M., Burke, M., *et al.* 2023. How open science can support the 3Rs and improve animal research. *Res. Ideas Outcomes* **9**: e105198. Pensoft Publishers.

Kambouris, S., Wilkinson, D.P., Smith, E.T. & Fidler, F. 2024. Computationally reproducing results from meta-analyses in ecology and evolutionary biology using shared code and data. *PLOS ONE* **19**: e0300333. Public Library of Science.

Kellner, K.F., Doser, J.W. & Belant, J.L. 2025. Functional R code is rare in species distribution and abundance papers. *Ecology* **106**: e4475.

Kidwell, M.C., Lazarević, L.B., Baranski, E., Hardwicke, T.E., Piechowski, S., Falkenberg, L.-S., *et al.* 2016. Badges to Acknowledge Open Practices: A Simple, Low-Cost, Effective Method for Increasing Transparency. *PLOS Biol.* **14**: e1002456. Public Library of Science.

Kim, B., Moran, N.P., Reinhold, K. & Sánchez-Tójar, A. 2021. Male size and reproductive performance in three species of livebearing fishes (Gambusia spp.): A systematic review and meta-analysis. *J. Anim. Ecol.* **90**: 2431–2445.

Kimmel, K., Avolio, M.L. & Ferraro, P.J. 2023. Empirical evidence of widespread exaggeration bias and selective reporting in ecology. *Nat. Ecol. Evol.* **7**: 1525–1536. Nature Publishing Group.

Kohrs, F.E., Auer, S., Bannach-Brown, A., Fiedler, S., Haven, T.L., Heise, V., *et al.* 2023. Eleven strategies for making reproducible research and open science training the norm at research institutions. *eLife* **12**: e89736.

König, L., Gärtner, A., Slack, H., Dhakal, S., Adetula, A., Dougherty, M., *et al.* 2025. How to bolster employability through open science. OSF.

Lai, J., Lortie, C.J., Muenchen, R.A., Yang, J. & Ma, K. 2019. Evaluating the popularity of R in ecology. *Ecosphere* **10**: e02567.

Maitner, B., Santos Andrade, P.E., Lei, L., Kass, J., Owens, H.L., Barbosa, G.C.G., *et al.* 2024. Code sharing in ecology and evolution increases citation rates but remains uncommon. *Ecol. Evol.* **14**: e70030.

Mandhane, P.J. 2024. Notice of Retraction: Hahn LM, et al. Post–COVID-19 Condition in Children. JAMA Pediatrics. 2023;177(11):1226-1228. *JAMA Pediatr.* **178**: 1085–1086.

Manzanedo, R.D., HilleRisLambers, J., Rademacher, T.T. & Pederson, N. 2021. Retraction Note: Evidence of unprecedented rise in growth synchrony from global tree ring records. *Nat. Ecol. Evol.* **5**: 1047–1047. Nature Publishing Group.

Markowetz, F. 2015. Five selfish reasons to work reproducibly. *Genome Biol.* **16**: 274.

Mills, J.A., Teplitsky, C., Arroyo, B., Charmantier, A., Becker, Peter.H., Birkhead, T.R., *et al.* 2015. Archiving Primary Data: Solutions for Long-Term Studies. *Trends Ecol. Evol.* **30**: 581–589.

Minocher, R., Atmaca, S., Bavero, C., McElreath, R. & Beheim, B. 2021. Estimating the reproducibility of social learning research published between 1955 and 2018. *R. Soc. Open Sci.* **8**: 210450. Royal Society.

Mislan, K. a. S., Heer, J.M. & White, E.P. 2016. Elevating The Status of Code in Ecology. *Trends Ecol. Evol.* **31**: 4–7. Elsevier.

Molloy, J.C. 2011. The Open Knowledge Foundation: Open Data Means Better Science. *PLOS Biol.* **9**: e1001195. Public Library of Science.

Müller, K. & Bryan, J. 2020. here: A Simpler Way to Find Your Files.

National Academies of Sciences, E., Affairs, P. and G., Committee on Science, E., Information, B. on R.D. and, Sciences, D. on E. and P., Statistics, C. on A. and T., *et al.* 2019. Reproducibility. In: *Reproducibility and Replicability in Science*. National Academies Press (US).

Parr, C.S. & Cummings, M.P. 2005. Data sharing in ecology and evolution. *Trends Ecol. Evol.* **20**: 362–363.

1087 Piwowar, H.A. & Chapman, W.W. 2010. Public sharing of research datasets: A pilot study of
1088 associations. *J. Informetr.* **4**: 148–156.

1089 Piwowar, H.A., Day, R.S. & Fridsma, D.B. 2007. Sharing Detailed Research Data Is Associated
1090 with Increased Citation Rate. *PLOS ONE* **2**: e308. Public Library of Science.

1091 Powers, S.M. & Hampton, S.E. 2019. Open science, reproducibility, and transparency in
1092 ecology. *Ecol. Appl.* **29**: e01822.

1093 Purgar, M., Klanjscek, T. & Culina, A. 2022. Quantifying research waste in ecology. *Nat. Ecol.*
1094 *Evol.* **6**: 1390–1397. Nature Publishing Group.

1095 R Core Team. 2022. R: A Language and Environment for Statistical Computing. R Foundation
1096 for Statistical Computing, Vienna, Austria.

1097 Reinecke, R., Trautmann, T., Wagener, T. & Schüler, K. 2022. The critical need to foster
1098 computational reproducibility. *Environ. Res. Lett.* **17**: 041005. IOP Publishing.

1099 Roche, D.G., Berberi, I., Dhane, F., Lauzon, F., Soeharjono, S., Dakin, R., *et al.* 2022. Slow
1100 improvement to the archiving quality of open datasets shared by researchers in ecology
1101 and evolution. *Proc. R. Soc. B Biol. Sci.* **289**: 20212780. Royal Society.

1102 Roche, D.G., Kruuk, L.E.B., Lanfear, R. & Binning, S.A. 2015. Public Data Archiving in Ecology
1103 and Evolution: How Well Are We Doing? *PLOS Biol.* **13**: e1002295. Public Library of
1104 Science.

1105 Rowhani-Farid, A., Aldcroft, A. & Barnett, A.G. 2020. Did awarding badges increase data
1106 sharing in BMJ Open? A randomized controlled trial. *R. Soc. Open Sci.* **7**: 191818. Royal
1107 Society.

1108 Sánchez-Tójar, A., Bezine, A., Purgar, M. & Culina, A. 2025. Code-sharing policies are
1109 associated with increased reproducibility potential of ecological findings. *Peer*
1110 *Community J.* **5**.

1111 Schneider, J., Rosman, T., Kelava, A. & Merk, S. 2022. Do Open-Science Badges Increase
1112 Trust in Scientists Among Undergraduates, Scientists, and the Public? *Psychol. Sci.* **33**:
1113 1588–1604. SAGE Publications Inc.

1114 Soeharjono, S. & Roche, D.G. 2021. Reported Individual Costs and Benefits of Sharing Open
1115 Data among Canadian Academic Faculty in Ecology and Evolution. *BioScience* **71**: 750–
1116 756.

1117 Tedersoo, L., Küngas, R., Oras, E., Köster, K., Eenmaa, H., Leijen, Ä., *et al.* 2021. Data sharing
1118 practices and data availability upon request differ across scientific disciplines. *Sci. Data*
1119 **8**: 192. Nature Publishing Group.

1120 Thrall, P.H., Chase, J., Drake, J., Espuno, N., Hello, S., Ezenwa, V., *et al.* 2023. From raw data
1121 to publication: Introducing data editing at Ecology Letters. *Ecol. Lett.* **26**: 829–830.

1122 Touchon, J.C. & McCoy, M.W. 2016. The mismatch between current statistical practice and
1123 doctoral training in ecology. *Ecosphere* **7**: e01394.

1124 Trisovic, A., Lau, M.K., Pasquier, T. & Crosas, M. 2022. A large-scale study on research code
1125 quality and execution. *Sci. Data* **9**: 60. Nature Publishing Group.

1126 Vazire, S. 2017. Quality Uncertainty Erodes Trust in Science. *Collabra Psychol.* **3**: 1.

1127 Viglione, G. 2020. 'Avalanche' of spider-paper retractions shakes behavioural-ecology
1128 community. *Nature* **578**: 199–200.

1129 Vines, T.H., Andrew, R.L., Bock, D.G., Franklin, M.T., Gilbert, K.J., Kane, N.C., *et al.* 2013.
1130 Mandated data archiving greatly improves access to research data. *FASEB J.* **27**: 1304–
1131 1308.

1132 Weissgerber, T.L., Gazda, M.A., Nilsonne, G., ter Riet, G., Cobey, K.D., Prieß-Buchheit, J., *et*
1133 *al.* 2024. Understanding the provenance and quality of methods is essential for
1134 responsible reuse of FAIR data. *Nat. Med.* **30**: 1220–1221. Nature Publishing Group.

1135 Wilkinson, M.D., Dumontier, M., Aalbersberg, Ij.J., Appleton, G., Axton, M., Baak, A., *et al.*
1136 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Sci.*
1137 *Data* **3**: 160018. Nature Publishing Group.

1138

Supplementary Material for The SORTEE Guidelines for Data and Code Quality Control in Ecology and Evolutionary Biology

Table S1. Summary table of the guidelines for each the six Stages of the SORTEE Guidelines for Data and Code Quality Control in Ecology and Evolutionary Biology

Stage	Guidelines
1. Data must be archived and adhere to FAIR guiding principles	
1.1 Data files are accessible and in an open repository	Data are open and freely available and are located in a permanent public repository with an associated globally unique persistent identifier, that is cited in the text and reference list of the manuscript.
1.2. Data are associated with a license	Data must be associated with a license.
1.3. Data files are present and complete	Authors must either provide: a) raw data, along with the processed data and/or code to prepare the data for analysis, or b) a sample of raw data alongside processed/filtered data when full raw data upload is not possible, or c) processed/filtered data with a detailed description of how to both obtain and process/filter the raw data.
1.4. Data files are in an interoperable format	Data files must be provided in an interoperable format.
1.5. Data metadata present and adequate	Detailed metadata, including (but not limited to) a README file, must accompany the data (see example in Figure 2).
2. Archived data corresponds with the data reported in the manuscript	The structure and contents of the archived data files must match the description in the manuscript.
3. Code must be archived and adhere to FAIR guiding principles	

3.1. Code files are accessible and in an open repository	Code files are open and freely available and are located in a permanent public repository with an associated globally unique persistent identifier, preferably in the same repository as the data files, and the repository is cited in the text and the reference list of the manuscript.
3.2. Code is associated with a licence	The code must be associated with a license.
3.3. Code files are present and complete	All code used for generating the results of the manuscript (including filtering, processing, graphing, and analysis) must be present either in one or more code files.
3.4. Code is in an interoperable format	Code files must be provided in an interoperable format.
3.5. Code metadata present and adequate	A sufficiently detailed README must accompany the code. Code must also be broken into sections with clear annotation stating the purpose of the code with clear links to the relevant sections, figures, and tables in the manuscript.
4. **Archived code corresponds with the workflow reported in the manuscript**	The structure and content of the archived code must match the description of data filtering, processing, and analysis, and the presentation of results in the manuscript.
5. **Archived code runs with the archived data**	Code must be able to run without error using the archived data. With the exception of easy to fix file path errors, all errors should be addressed by the author.
6. Results can be computationally reproduced by running the archived code	Results reproduced by the data editor with the archived data and code must match those presented in the manuscript. A tolerance threshold can be given when there is not an exact match but the authors must state clearly in the code metadata why this mismatch might occur. If saved model outputs are instead provided, this must also be clearly stated in the metadata.