Author response to reviews of

WORCS: A Workflow for Open Reproducible Code in Science

Caspar J. Van Lissa, Andreas M. Brandmaier, Loek Brinkman, Anna-Lena Lamprecht, Aaron Peikert, Marijn E. Struiksma, & Barbara Vreede
submitted to *Data Science*

RC: ***Reviewer Comment*** AR: Author Response ☐ Manuscript text

Dear Dr. Kuhn,

We would like to thank you for providing us with the opportunity to revise our manuscript, "WORCS: A Workflow for Open Reproducible Code in Science". We sincerely appreciate the time you and the Reviewers have taken to read the manuscript, and the extensive and thought-provoking comments you have raised. We hereby submit our revision of the manuscript for your consideration. We have attempted to address each of the issues raised by yourself and the Reviewers on a point-by-point basis, and detail the changes made in the action letter below. We would like to thank you and the Reviewers for your time, and we look forward to receiving your response.

Sincerely, on behalve of all coauthors, Caspar van Lissa

1. Reviewer #1

Review #1 submitted on 19/Oct/2020 by Daniel Garijo ORCID logo `https://orcid.org/0000-0003-0454-7145`

- Overall Impression: Weak
- Suggested Decision: Reject
- Technical Quality of the paper: Good
- Presentation: Good
- Reviewer's confidence: High
- Significance: High significance
- Background: Reasonable
- Novelty: Limited novelty
- Data availability: All used and produced data (if any) are FAIR and openly available in established data repositories
- Length of the manuscript: The length of this manuscript is about right

This paper describes worcs, a package and workflow to help make R code reproducible according to the TOP guidelines.

The paper describes a tutorial of the tool, and how it implements the TOP guidelines.

Reasons to accept:

- The paper is relevant to the journal and on a timely topic.
- The capabilities of the tool for synthetic data generation, connect to OSF, preprint publishing and dependency gathering show a lot of promise.
- All the software (except a 404 page in the help, which is minor) is available online; and the project seems maintained and well documented.

Reasons to reject:

RC: C1 There is no user evaluation, and therefore all the claims about worcs are unsupported by evidence. If the authors had included a compelling evaluation I would be in favor of accepting the paper.

AR: Response: We appreciate the recommendation of performing a user evaluation, and this is certainly a next step in the development of WORCS. At this point, we have written this manuscript detailing the conceptual workflow, and released a software implementation for R-users. The next step, which we are now entering, is to build a user base, and then mount a user evaluation.

To address the Reviewer's comment, we have set up an evaluation questionnaire, and sent it to the attendees of our previous workshops on WORCS. At the time of writing, responses are coming in. These are generally very favorable, and have not yet revealed major points of improvement.

We further want to point out that we have a strong history of addressing user comments submitted through the GitHub issues page. WORCS adheres to the Core Infrastructure Initiative "best practices" (CII Best Practices), which requires that the channels through which users can provide feedback are clearly indicated, as they are here. It is also evident from the resolved tickets on GitHub that we have been very responsive in actively incorporating user feedback. This is another requirement of CII Best Practices.

We are committed to continue evaluating user experience, both actively through user evaluation forms, and passively through the GitHub issues pages. We hope that this commitment, along with the evidence of our prior responsiveness to user feedback presented above, help assuage the Reviewer's concerns.

Finally, we wish to point out that the Editor has indicated that 'resource papers do not need to have a full-blown user evaluation, but we expect "sound evidence of its (potential for) reuse" '. We therefore also elaborate on the efforts we have made to ensure that WORCS is reusable, followed by evidence of its current adoption rates.

We have made the following efforts to ensure potential for reuse:

1. WORCS is implemented in an R-package, and thus readily available to R-users
2. This R-package is available in a standardized repository, namely CRAN ("The Comprehensive R Archive Network"), and can thus be installed through default channels
3. The software meets all criteria of the Core Infrastructure Initiative "best practices" (CII Best Practices), including clearly indicated channels for user feedback, and being responsive to that feedback
4. The R-package uses only tools that are fully integrated within the RStudio IDE, thus enhancing ease of use
5. The R-package is compatible with other reproducibility solutions, including Docker, drake, and make

There is also evidence that WORCS is already being adopted, even prior to publication (as of 14-11-2020):

1. The preprint has been downloaded 582 times, since being published on 31-05-2020

2. The `worcs` R-package has been downloaded 2567 times from CRAN, since being published on 18-05-2020
3. The GitHub project has been forked 8 times, watched 5 times, and starred 40 times
4. The lead author has given invited lectures on WORCS at:
    - Max-Planck-Institute for Human Development (2020)
    - Open Science Community Rotterdam (2020)
    - Eindhoven University of Technology (2020)
5. WORCS is currently used in the following GitHub repositories (sorted by user):
    - cjvanlissa
        - veni_sysrev
        - IHBT-SEM
        - schumpe
        - storm_meta_father_mother
        - placebo_adhd
        - mpib
        - coping_covid
    - IndrajeetPatil
        - worcs_practice
    - EstherPlomp
        - RDM-survey
    - lisallreiber
        - worcs_ws_empty
    - KimBaldry
        - BIO-MATE

Assignment: **CJ, BV, AL**

Done

RC: C2 This paper describes worcs, a package and workflow to help make R code reproducible according to the TOP guidelines.

The paper is well written, easy to follow and very relevant and timely for the journal. I think the capabilities of the worcs package for synthetic data generation, connect to osf, help with the preprints and capture dependencies are very valuable to the community. However, the main limitation of the paper is that the workflow and proposed package are not evaluated with users, and therefore the claims of the authors with respect of its capabilities are not supported by evidence. I would like to encourage the authors to gather user/community feedback and demonstrate with numbers that worcs can help scientists as they claim. Then, the paper would become a strong contribution for the journal.

AR: Response: In addressing this comment, we respectfully refer to our response to the preceding comment ("There is no user evaluation", R1 C1), and again emphasize the Editor's comment that "resource papers do not need to have a full-blown user evaluation". We would also like to clarify that the goal of the present manuscript is to define the problem WORCS intends to address (specifically; meeting the TOP-guidelines for Open Science), introduce the tools WORCS uses to address this problem, explain how WORCS uses these tools, and provide a step-by-step workflow. It is not clear what "claims" are made that are unsupported by evidence. In the revision, we have attempted to further clarify this goal from the paper onset:

Abstract:

> This paper introduces the Workflow for Open Reproducible Code in Science (WORCS): A step-by-step procedure that researchers can follow to make a research project open and reproducible. The purpose of the workflow is to lower the threshold for adoption of open science principles. It is based on established best practices, and can be used either in parallel to, or in absence of, top-down requirements by journals, institutions, and funding bodies. To facilitate widespread adoption, the WORCS principles have been implemented in the R package `worcs`, which offers an RStudio project template and utility functions for specific workflow steps. This paper introduces the conceptual workflow, discusses how it meets different standards for open science, and addresses the functionality provided by the `R` implementation, `worcs`.

We again emphasize that we take the recommendation to perform user evaluation very seriously, but we also feel that it would be premature to do so even before the paper is published. We have built WORCS based on established tools and best practices, and we have gathered preliminary positive feedback from the community through workshops and teaching.

Assignment: **CJ**

Done

RC: C3 Unsupported claims (beyond the ones derived from an absence of evaluation): The paper claims in many parts that worcs helps make an entire research project Open and Reproducible. However, at the same time there is also a claim for not supporting replication. Since replication is generally understood as a necessary steps towards reproducibility, I find this kind of contradictory. Maybe the claims can be modified stating that worcs helps documentation and understanding, hence supporting FAIR.

AR: Response: We thank the Reviewer for bringing this apparent contradiction to our attention. The misunderstanding arises from a lack of explicit definitions of the terms "replication" and "reproducibility" in the original manuscript. We are aware of the oft-lamented lack of consistent definitions for these terms (e.g., Patil, Peng, & Leek, 2019; Plesser, 2018). In the revision, we have attempted to address this lack of clarity by including explicit definitions for these terms:

> Throughout this paper, we adhere to the definition of open science as "the practice of science in such a way that others can collaborate and contribute, where research data, lab notes and other research processes are freely available, under terms that enable reuse, redistribution and reproduction of the research and its underlying data and methods" (Vicente-Saez and Martinez-Fuentes 2018).
> [...] reproducibility can be defined as "re-performing the same analysis with the same code and the same data using a different analyst", and replicability as "re-performing the experiment and collecting new data" (Patil, Peng, and Leek 2019).

After making these definitions explicit, it becomes clear that the Reviewer's definition of replication as "a necessary steps towards reproducibility" is not consistent with the definitions used in the present paper.

We further wish to clarify that the fact that we address guidelines 1-7, but not guideline 8 ("replication") is fully consistent with the intended purpose of the TOP-guidelines. Specifically, the first seven guidelines are intended to ensure the "reproducibility of the reported results based on the originating data" (direct quote from Nosek et al. 2015). The eighth criterion, by contrast, is "not formally a transparency standard for authors, [but] addresses journal guidelines for consideration of independent replications for publication" (again a direct quote). To avoid further confusion or controversy, we now address this distinction between the first 7 guidelines and the final 8th explicitly in the revised manuscript:

> These guidelines describe eight standards for open science, the first seven of which "account for reproducibility of the reported results based on the originating data, and for sharing sufficient information to conduct an independent replication" (Nosek et al. 2015). These guidelines are: 1) Comprehensive citation of literature, data, materials, and methods; 2) sharing data, 3) sharing the code required to reproduce analyses, 4) sharing new research materials, and 5) sharing details of the design and analysis; 6) pre-registration of studies before data collection, and 7) pre-registration of the analysis plan prior to analysis. The eighth criterion is "not formally a transparency standard for authors", but "addresses journal guidelines for consideration of independent replications for publication". In this context, reproducibility can be defined as "re-performing the same analysis with the same code using a different analyst", and replicability as "re-performing the experiment and collecting new data" (Patil, Peng, and Leek 2019). WORCS defines the goals of open science in terms of the first seven TOP-guidelines, and the workflow and its `R` implementation are designed to facilitate compliance therewith.

Assignment: **CJ, ALL**

Done

RC: C4: I am confused by the citation "Van Lissa, Caspar J., Aaron Peikert, and Andreas M. Brandmaier. 2020. "Worcs: Workflow for Open Reproducible Code in Science."". Is this referring to another paper or the actual package? If it's another paper then what is the contribution of this publication?

AR: Response: We apologize; it appears that the reference list was not parsed correctly after rendering the manuscript to HTML - the preferred format of Data Science. The reference is indeed to the R-package. In the revised manuscript, we have repaired the references so that it is clear that this is a software package.

Assignment: **CJ**

Done

RC: C5: If the paper is generated as DDG, where are the functions that show it's capabilities? It is also not clear to me that some of the functionalities of Latex would be usable in RMarkdown, such as all the equations.

AR: Response: We understand that our introduction of the RMarkdown format was a bit too brief, and that readers may wonder whether specific functionalities of LaTeX would be usable in RMarkdown. To address this comment, the revised manuscript now reads:

> In the `R` implementation of WORCS, we recommend centering a research project around one dynamically generated `RMarkdown` document (Xie, Allaire, and Grolemund 2018). RMarkdown is based on `Markdown`, which uses plain-text commands to typeset a document. It enhances this format with support for R-code, and is compatible with LaTeX math typesetting. For a full overview of the functionality of `RMarkdown`, see Xie, Allaire, and Grolemund (2018).

Assignment: **CJ**

Done

RC: C6: It is not clear how worcs addresses depedency management. The section lists a series of tools that can be used, but the section is not very specific to worcs.

AR: Response: We apologize for the apparent lack of clarity of this section. We have restructured and clarified this section to convey the following information:

1) Introduce the general concept of dependency management
2) Introduce `renv`, the solution for dependency management used by the R-implementation of `worcs`
3) Acknowledge the more comprehensive approach to dependency management offered by Docker

The changes are extensive, so we will not quote them all here. Instead, we refer the Reviewer to the revised manuscript.

We have also, in response to another Reviewer comment, created a Docker Hub build that runs `worcs`. Therefore, as of this revision, both of the tools discussed in this section (i.e., `renv` and Docker) are directly relevant for WORCS-users, and the section should read less like a "series of tools that can be used". If the Reviewer so desires, we would also be willing to cut the brief discussion of Code Ocean - but this had been included at the request of an informal reviewer of this paper (Daniël Lakens).

Assignment: **CJ**

Done

RC: C7a: The workflow of releasing data products on GitHub is not ideal. First, because datasets of 100 MBs usually require special handling. [...] The authors seem to not describe very well what would happen in data science experiments where the data size is significant.

AR: Response: We have split this comment up to separately address the handling of large data, and findability (see below).

With regard to the handling of large data, we agree that GitHub is not an ideal platform for handling data larger than 100MB. It is important to note, however, that there are no restrictions for sharing files up to that

size. Further, take into consideration that 100MB is very large for tabular data; for example, if only integer numbers are stored that is $(100MB * 1024^2)/(2^2) = 26214400$ unique values. In the social sciences, datasets rarely exceed this limit - and so for our readership, GitHub is as good a platform for sharing data as any. Moreover, GitHub is ideal for version controlling analysis code, RMarkdown documents, and small to medium-sized data files. In our opinion, the advantages of having **all** of a paper's resources in the same repository outweighs the advantages of encouraging every reader to additionally use specialized solutions for "big data" (which most will not need).

To address this comment, we now explicitly address the problem of version controlling large files, and introduce a solution for larger files, which can be linked to git using Git Large File Storage:

> Data sharing may be impeded by several concerns. One technical concern is whether data require special storage provisions, as is often the case with "big data". Most Git remote repositories do not support files larger than 100MB, but it is possible to enhance a Git repository with Git Large File Storage to add remotely stored files as large as several GB. Files exceeding this size require custom solutions beyond the scope of this paper.

We also added a paragraph to the Limitations section that discusses the fact that no single workflow is suitable for all situations:

> Another limitation is the fact that no single workflow can suit all research projects. Not all projects conform to the steps outlined here; some may skip steps, add steps, or follow similar steps in a different order. In principle, WORCS does not enforce a linear order, and allows extentensive user flexibility. A related concern is that some projects might require specialized solutions outside of the scope of this paper. For example, some projects might use very large data files, rely on data that reside on a protected server, or use proprietary software dependencies that cannot be version controlled using `renv` or Docker. In such cases, the workflow can serve as a starting point for a custom approach; offering a primer on best practices, and a discussion of relevant factors to consider. We urge users to inform us of such challenges and custom solutions, so that we may add them to the overview of WORCS-projects on GitHub, where they can serve as an example to others.

Assignment: **BV, AB, AL, CJ**

Done

RC: C7b: Second, it doesn't follow very well the FAIR principles for findability and attribution of data. Instead, repositories like Zenodo or FigShare could be user to address data storage, with proper meta-data and its corresponding citation.

AR: Response: We agree completely that it is important to make open data findable beyond just providing it freely. In order to address the Reviewer's comment, we've revised the paper and the workflow vignette to emphasize more clearly that an essential step of the workflow is to connect the GitHub repository to a project page on the Open Science Framework - a standard repository for open research. We also devote more attention to the importance of DOIs; both for the project as a whole, and for specific resources, such as data files. Finally, we have elaborated on the discussion of the FAIR principles, emphasizing that WORCS is not designed to address these comprehensively, but pointing out specific ways in which WORCS is consistent with these guiding principles. Please see our response to Reviewer 2, comment 2 for more details.

With regard to the specific platforms named by the Reviewer: We are not aware of arguments to support a strong preference for Zenodo or FigShare over GitHub. Without such arguments, as far as we are concerned, these repositories are largely interchangeable. One key advantage of storing the data on GitHub is that they are then bundled with the documentation (e.g., codebook), analysis code, and manuscript. Thus, readers have access to **all** materials required for replication in one repository. Moreover, the entire project can be readily updated from the console, and is version controlled. GitHub additionally offers user interaction with the materials through forking, cloning, opening issues, and sending pull requests - all useful tools for replication and collaboration.

It is further worth noting that GitHub is compatible with Zenodo; users can document GitHub repositories in Zenodo and generate a DOI. We now explicitly mention this option in the paper and workflow vignette. We also mention the potential redundancy with Steps 15-16 of the workflow, however: Connecting the GitHub repository to an OSF project and generating a DOI accomplishes effectively the same.

Assignment: **BV, AB, AL, CJ**

Done

RC: C8: In the 3 phases listed, it is not clear what worcs does and does not for users.

AR: Response: It is not entirely clear to us whether the Reviewer is referring to the conceptual workflow ("WORCS"), or to the `R` implementation of WORCS, i.e., `worcs`.

We suspect that the Reviewer means the `worcs` R-package, so to address this comment, the Revision now states the following:

> We refer users of the `worcs` implementation in `R` to the workflow vignette, which describes in more detail which steps are automated by the `R` package and which actions must be taken by the user. The vignette is also updated with future developments of the package.

Just in case we misunderstood, and the Reviewer actually meant the conceptual workflow WORCS, we have also added a few sentences to clarify the purpose of the conceptual workflow:

> We provide a conceptual outline of the WORCS procedure below, based on WORCS Version 0.1.6. The conceptual workflow recommends actions to take in the different phases of a research project, in order to work openly and transparently in compliance with the TOP-guidelines and best practices for open science. Note that, although the steps are numbered for reference purposes, we acknowledge that the process of conducting research is not always linear.

Assignment: **CJ**

Done

RC: C9: In some cases it looks like the authors claim that everyone should use R for open science. While R has a strong community behind, I believe there are others with very strong support (e.g. Python). I am not sure if suggesting a shift to R is the right approach. Note that this is the impression I got from reading the paper, maybe it was unintended by the authors.

AR: Response: We wish to emphasize that it was certainly unintended to suggest that everyone should us R for open science. To address this comment, we have done two things: First, throughout the paper, we have

attempted to clarify the distinction between the platform independent conceptual workflow "WORCS", and its implementation for R users, `worcs`. Second, throughout the paper, we encourage the implementation of WORCS in other languages. Most illustrative of these changes is the start of the paragraph on **The R implementation of WORCS**:

> WORCS is, first and foremost, a conceptual workflow that could be implemented in any software environment. As of this writing, the workflow has been implemented for `R` users in the package `worcs` (Van Lissa, Peikert, and Brandmaier 2020). Several arguments support our choice to implement this workflow first in `R` and RStudio, although we encourage developers to port the software to other languages.

Assignment: **CJ**

Done

RC: C10: Vignete on citation leads to a 404.

AR: Response: We apologize for the oversight; we have fixed this URL and checked all URLs in the manuscript for mistakes (none found).

Assignment: **CJ**

Done

RC: C11: Synthetic data generation is fantastic, but it is not clear to me how does worcs actually keep the data consistent e.g. for training models. Does it follow a similar distribution to the source data?

AR: Response: We agree that synthetic data generation is exciting, but feel obliged to clarify that this is **not** a novel contribution of the present paper nor R-package. As indicated in the paper, we use the algorithm by Nowok and colleagues, who explain the details of the method. We re-implemented the method in the `worcs` package because the `synthpop` package is rarely maintained (e.g., one of our pull requests took almost 10 months to be accepted), and because we wanted a more flexible, customizable interface.

To answer the Reviewer's specific question, the synthetic data indeed follow similar distributions to the source data, as indicated in the paper:

> Synthetic data mimic the level of measurement and (conditional) distributions of the real data (see Nowok, Raab, and Dibben 2016).

We have additionally clarified the paragraph that introduces the `synthetic()` function, and added further references to the methodological paper, and to the documentation of the `synthetic()` function (which summarizes the algorithm).

Assignment: **CJ**

Done

RC: C12: Docker is deemed as complicated for novice users, but the setup vignete looks quite complex to me as well. Wouldn't a Docker container with worcs installed be easier to use? Maybe these aspects would be better highlighted in a user evaluation.

AR: Response: We thank the Reviewer for this thoughtful suggestion. To address this comment, we have published an image on Docker Hub with all requirements of `worcs` installed, and added a Docker-specific setup vignette to the package and website, which is referenced in the Setup pagragraph:

> Before you can use the `R` implementation of the WORCS workflow, you have to install the required software. This 30 minute procedure is documented in the setup vignette. After setting up your system, you can use `worcs` for all of your projects. Alternatively, Docker users can follow the stetup with Docker vignette.

Assignment: **AP, AB**

Done

2. Reviewer #2

Submitted on 19/Oct/2020 by Remzi Celebi ORCID logo `https://orcid.org/0000-0001-7769-4272`

- Overall Impression: Good
- Suggested Decision: Accept
- Technical Quality of the paper: Good
- Presentation: Good
- Reviewer's confidence: Medium
- Significance: High significance
- Background: Reasonable
- Novelty: Limited novelty
- Data availability: With exceptions that are admissible according to the data availability guidelines, all used and produced data are FAIR and openly available in established data repositories
- Length of the manuscript: The length of this manuscript is about right

The paper introduces the Workflow for Open Reproducible Code in Science (WORCS) , a workflow that complies with most of best practices for open sciences projects and demonstrates the use of the R worcs package for adapting WORCS workflow. The package provides a template for R users to create projects that follow open science principles.

Reasons to accept:

It provides a good discussion of TOP-guidelines (a set of open science principles) and how proposed workflow/package can be used to facilitate the adaptation of these guidelines.

Reasons to reject:

RC: C1: I would suggest the authors include a flowchart for users that shows the decision making process for which tools/operations should be used in WORCS. An example project would also help users better understand the package.

AR: Response: We thank the Reviewer for these suggestions. Regarding the flowchart - we had already included one in the previous version of the manuscript. To address this comment, we have clarified the references in the text to this flowchart:

> The WORCS workflow constitutes a step-by-step procedure that researchers can follow to make a research project open and reproducible. These steps are elaborated in greater detail below. See Figure 1 for a flowchart that highlights important steps in different stages of the research process.

Regarding the example project, we have now curated a list of all public `worcs` projects on GitHub, and we reference this list in several places in the manuscript, including:

Abstract

> The source code for the `R` package and manuscript, and a list of user examples of WORCS projects, are available at `https://github.com/cjvanlissa/worcs`.

Introducing the workflow

> For examples of how the workflow is used in practice, we maintain a list of public `worcs` projects on the WORCS GitHub page.

Assignment: **CJ**

Done

RC: C2: The claim that WORCS is amenable to the FAIR Guiding Principles should be justified. What are the solutions provided by WORCS that make a project interoperable and reusable? It would be improved with a more thorough discussion on how WORCS meets each FAIR principle.

AR: Response: To address this comment, we have made three changes:

1) We have changed the workflow, emphasizing the importance of creating a project on the OSF and connecting it to the GitHub repository, and emphasizing the importance of DOIs and a license
2) We have rewritten the paragraph addressing the FAIR principles, with a more thorough discussion of how WORCS can help address each FAIR principle. We also state explicitly that *WORCS was not designed to comprehensively address these principles, but the workflow does facilitate meeting them.*
3) We have made minor changes to the `worcs` package, e.g., to include information on the project readme about how readers can obtain access to data.

This paragraph now reads:

Aside from the TOP Guidelines, the FAIR Guiding Principles for scientific data management and stewardship (Wilkinson et al. 2016) are an important standard for open science. These principles advocate that digital research objects should be Findable, Accessible, Interoperable and Reusable. Initially mainly promoted as principles for data, they are increasingly applied as a standard for other types of research output as well, most notably software (Lamprecht et al. 2019).
WORCS was not designed to comprehensively address these principles, but the workflow does facilitate meeting them. In terms of Findability, all files in GitHub projects are searchable by content and meta-data, both on the platform and through standard search engines. The workflow further recommends that users create a project page on the Open Science Framework – a recognized repository in the social sciences. When doing so, it is important to generate a Digital Object Identifier (DOI) for this project. A DOI is a persistent way to identify and connect to an object on the internet, which is crucial for findability. Additional DOIs can be generated through the OSF for specific resources, such as data sets. It is further worth noting that, optionally, GitHub repositories can be connected to Zenodo: On Zenodo, users can store a snapshot of the repository, provide metadata, and generate a DOI for the project or specific resources. When a project is connected to the OSF, as recommended, these steps may be redundant. Finally, each worcs project has a [YAML file] (`https://yaml.org/`) that lists the data objects (and other meta-data) in the project, which will allow web crawlers to index worcs projects. We use this metadata, for example, to identify public `worcs` projects on GitHub.
Accessibility is primarily safeguarded by hosting all project files in a public GitHub repository, which can be downloaded as a ZIP archive, and cloned or forked using the Git protocol. GitHub is committed to long term accessibility of these resources. We further encourage users to make their data accessible when possible. When data must be closed, the `worcs` function `closed_data()` adds a paragraph to the project readme, indicating how readers can contact the author for access.
With regard to interoperability, the workflow recommends using plain-text files whenever possible, which ensures that code and meta-data are human- and machine-readable. Furthermore, the `worcs` package is implemented in R, which is open source. To promote reusability, we recommend users to specify an appropriate license for their projects. The `worcs` package includes the CC licenses, and suggests a CC-BY 4.0 license by default. This license is featured in the GitHub repository, and we encourage users to also display it on their project's OSF page. To further facilitate reusability, `worcs` automatically generates a codebook for each data file. We encourage users to elaborate on the default codebook by adding variable description and categories, which would - in the future - enable indexing data by topic area.

Assignment: **AL, BV, CJ**

Done

RC: C3: Is WORCS providing any service for de-identification?

AR: Response: WORCS does not provide a service for de-identification, and we consider this to be outside of the scope of the present paper. If the Reviewer feels differently, we are open to being convinced, however!

The current team lacks relevant expertise to make this contribution, but since `worcs` is open source software, anybody is welcome to contribute this functionality. Such a contribution would warrant coauthorship on the package, in line with the contribution guidelines outlined here.

Assignment: **ALL**

Done

3. Reviewer #3

Submitted on 26/Oct/2020 by Anonymous

- Overall Impression: Weak
- Suggested Decision: Reject
- Technical Quality of the paper: Average
- Presentation: Good
- Reviewer's confidence: High
- Significance: High significance
- Background: Reasonable
- Novelty: Limited novelty
- Data availability: All used and produced data (if any) are FAIR and openly available in established data repositories
- Length of the manuscript: The length of this manuscript is about right

The paper presents WORCS, a library that is implemented within R-studio with the purpose of promoting the creation of reproducible research. In particular, they assist the researchers in the tasks of generating the scholarly article, thereby managing a number of tasks that researchers have to deal with from the reformulation of their hypothesis to version control, to dependency management, and finally generating the scholarly article.

Reasons to accept:

On the positive side, the authors built their tool, with Open Science principles in mind. In doing so, they have pined the actions that need to be performed as part of the research process. For instance, steps such as the pre-registration of the objective of the study before starting the investigation is ignored by existing solutions.

Reasons to reject:

The objective of the author is ambiguous. That said, I do not think that WORCS provide sufficient features that make it stands out from state of the art tools, for the following reasons.

RC: C1: The process as described by the author for conducting research is assumed to be linear. However, researchers often go back and forth on steps that they have already completed, and sometimes have to undo them completely and start from scratch. This does not seem to be tackled by the tool.

AR: Response: We agree with the Reviewer that researchers often go back and forth in their work. We did not intend to convey the impression that the process of conducting research is assumed to be linear. Of course, since we are presenting a workflow - there are a number of steps involved, which we have placed in roughly chronological order and numbered for reference purposes. For example, Study Design must logically occur prior to Writing and Analysis, and Publication can only logically occur after Writing. We also wish to emphasize that the "tool" (we assume the Reviewer is referring to the `worcs` R-package) does not enforce any kind of linearity.

To address the Reviewer's comment, we have made two changes: First, we now explicitly state, in two places in the paper, and in the Workflow vignette included with the `worcs` package, that the steps are not necessarily linear:

Introducing the workflow:

> Note that, although the steps are numbered for reference purposes, we acknowledge that the process of conducting research is not always linear.

Limitations:

> Another limitation is the fact that no single workflow can suit all research projects. Not all projects conform to the steps outlined here; some may skip steps, add steps, or follow similar steps in a different order. In principle, WORCS does not enforce a linear order, and allows extentensive user flexibility.

Second, while it was always possible to manually add a manuscript or preregistration at a later point in time after creating a new project in RStudio - we have added more functions to increase the non-linear flexibility of the `worcs` package: `add_manuscript()` and `add_preregistration()`.

Assignment: **CJ**

Done

RC: C2: The gist of the tool seems to allow researchers to document their steps and generate a scholarly document that is ready for submission. In a sense, there is a similarity with existing notebooks such as Jupyter. So I was wondering why don't the authors start from an existing popular tool such as Jupyter as the starting point (given the similarity in features that they would like to add), and extend Jupyter for instance with the new features such as dependency documentation and version control ?

AR: Response: There are two points we wish to address here. First, the Reviewer focuses on "the tool" - which we assume refers to the R package. However, the main goal of this paper is to introduce the conceptual workflow. The R package is merely offered to make adoption of the conceptual workflow easier for R users. This distinction is clarified several places throughout the manuscript, and to prevent further confusion, we have clarified the distinction even further in the revision, e.g.:

Abstract:

> This paper introduces the Workflow for Open Reproducible Code in Science (WORCS): A step-by-step procedure that researchers can follow to make a research project open and reproducible.
> [...]
> To facilitate widespread adoption, the WORCS principles have been implemented in the R package `worcs`, which offers an RStudio project template and utility functions for specific workflow steps.

Manuscript, introduction:

> This paper introduces the conceptual workflow (referred to in upper case, "WORCS"), and discusses its underlying principles and how it meets different standards for open science. The principles underlying this workflow are universal and largely platform independent. We also present a software implementation of WORCS for `R` users (R Core Team 2020): The `R` package `worcs` (referred to in monospace font, Van Lissa, Peikert, and Brandmaier 2020).

Manuscript, paragraph titled **The R implementation of WORCS**:

> WORCS is, first and foremost, a conceptual workflow that could be implemented in any software environment. As of this writing, the workflow has been implemented for `R` users in the package `worcs` (Van Lissa, Peikert, and Brandmaier 2020).

Second, in response to the question "why don't the authors start from an existing popular tool such as Jupyter", we completely agree with the Reviewer that the conceptual workflow as explained in the paper could be implemented in any software environment, including Jupyter. In fact, we emphasize this in several places throughout the manuscript, for instance the paragraph titled **The R Implementation of WORCS**:

> WORCS is, first and foremost, a conceptual workflow that could be implemented in any software environment. As of this writing, the workflow has been implemented for `R` users in the package `worcs` (Van Lissa, Peikert, and Brandmaier 2020). Several arguments support our choice to implement this workflow first in `R` and RStudio, although we encourage developers to port the software to other languages.

And in **Future developments**:

> although the workflow is currently implemented only in `R`, it is conceptually relevant for users of other statistical programming languages, and we welcome efforts to implement WORCS in other platforms."

To answer the Reviewer's question directly: We implemented this workflow in R because we are R developers. But in the paper, we also provide some arguments that compel our preference for R and RStudio as a choice for statistical computing in open science:

First, R and all of it extensions are free open source software, which make it a tool of choice for open science. Second, all tools required for an open science workflow are implemented in R, most of these tools are directly accessible through the RStudio user interface, and some of them are actively developed by the team behind RStudio. Third, unlike any competitor, RStudio is a Public Benefit Corporation, whose primary purpose is explicitly aligned with open science principles:
RStudio's primary purpose is to create free and open-source software for data science, scientific research, and technical communication. This allows anyone with access to a computer to participate freely in a global economy that rewards data literacy; enhances the production and consumption of knowledge; and facilitates collaboration and reproducible research in science, education and industry.
Fourth, R is the second-most cited statistical software package (Muenchen 2012), following SPSS, which has no support for any of the open science tools discussed in this paper. Fifth, R is well-supported by a vibrant and inclusive online community, which develops new methods and packages, and provides support and tutorials for existing ones. Finally, R is highly interoperable: Packages are available to read and write nearly every filetype, including DOCX, PDF (in APA style), and HTML. Moreover, wrappers are available for many tools developed in other programming languages, and code written in other programming languages can be evaluated from R, including C++, Fortran, and Python (Allaire et al. 2020). There are excellent free resources for learning R (e.g., Grolemund and Wickham 2017).

Assignment: **BV, CJ**

Done

RC: C3: Regarding Version control and dependency documentation, the authors uses in their system existing solution, namely git and env, respectively. While I did not criticize reusing existing solution, on the contrary, I don't believe that simply using such solution add a substantial value. For example, git capture changes at the level of the line of code/text, whereas a researcher needs an abstraction that informs him/her on the coarse-grained elements that are significant from the point of view of development and research steps so as to have a clear idea of where s/he is within the process and help him/her make the decision of the next steps to undertake.

AR: Response: Respectfully, we are not entirely sure what the criticism is here. First, as addressed in our response to the previous comment, there seems to be a misunderstanding about the scope of the paper. To clarify, we again state that our paper introduces a workflow for open science, based on objective criteria set out in the TOP-guidelines. Regarding novelty: Aside from the work-in-press of our co-authors - which is focused on strict computational reproducibility and is compatible with the present workflow - this is one of the first efforts to translate the requirements of open science into a step-by-step workflow. When you search Google Scholar for "open science workflow", the first hit that actually describes a workflow is the present manuscript.

Second, the Reviewer claims that using e.g. Git does not add substantial value - but this claim is in disagreement with published literature (which we cite) that makes a strong case for the value of using Git for research (see Ram 2013; Blischak, Davenport, and Wilson 2016).

Third, there seems to be some misunderstanding of the workings of existing solutions. For example, the Reviewer remarks that "git capture changes at the level of the line of code/text, whereas a researcher needs an abstraction that informs him/her on the coarse-grained elements [...] so as to have a clear idea of where

s/he is within the process and help him/her make the decision of the next steps to undertake". This comment suggests a fundamental misunderstanding of how Git works. Although Git tracks file changes on a line-by-line basis, such changes are typically grouped together in a "commit", which is how Git captures the more "course-grained elements" the Reviewer refers to. For example, in our response to these reviews, we have created one Git commit for each Reviewer comment, and these commits contain changes to the action letter, manuscript, and vignettes.

To address the Reviewer's comment, we now clarify this point in the revised manuscript:

> Git tracks changes to files on a line-by-line basis. The user can store these changes by making a "commit": a snapshot of the version controlled files. Each "commit" can contain as many changes as desired; for example, a whole new paragraph, or many small changes made to address a single reviewer comment. Each commit is further tagged with a message, describing the changes. The `R` implementation of `worcs` contains a convenience function, `git_update("your commit message")`, which creates a new commit for all changes since the previous commit, labels it with "your commit message", and pushes these changes to the remote repository. It is, effectively, a Git "quick save" command.

The GitHub function of creating "releases" also helps mark the "course-grained" progress of a research project. In the WORCS procedure, we encourage taggin the preprint and submitted version of a manuscript as a "release" on GitHub. Such releases are prominently featured on the GitHub project page, and are accompanied by a downloadable archive of the historic state of the project. To further address the Reviewer's comment, we now mention this in the manuscript text as well:

> The GitHub platform offers additional functionality over Git. One such feature is that specific stages in the lifecycle of a project can be tagged as a "release": A named downloadable snapshot of a specific state of the project, that is prominently featured on the GitHub project page. The WORCS procedure encourages users to create releases to demarcate project milestones such as preregistration and submission to a journal. Releases are also useful to mark the submission of a revised manuscript, and publication.

Assignment: **CJ**

Done

RC: C4: To sum up, I think that the initial idea is interesting. However, I do not think that WORCS will be adopted by researchers or promote the state of the art of reproducibility, even if we focus on researchers who conduct their development and analysis using R.

AR: Response: We thank the Reviewer for acknowledging the idea to be interesting. Whether it will be adopted by researchers remains to be seen after publication. We can offer some prelimiary evidence that the workflow is already being adopted (as of 14-11-2020):

1. The preprint has been downloaded 582 times, since being published on 31-05-2020
2. The `worcs` R-package has been downloaded 2567 times from CRAN, since being published on 18-05-2020
3. The GitHub project has been forked 8 times, watched 5 times, and starred 40 times

4. The lead author has given invited lectures on WORCS at:
    - Max-Planck-Institute for Human Development (2020)
    - Open Science Community Rotterdam (2020)
    - Eindhoven University of Technology (2020)

5. WORCS is currently used in the following GitHub repositories (sorted by user):
    - cjvanlissa
        - veni_sysrev
        - IHBT-SEM
        - schumpe
        - storm_meta_father_mother
        - placebo_adhd
        - mpib
        - coping_covid
    - IndrajeetPatil
        - worcs_practice
    - EstherPlomp
        - RDM-survey
    - lisallreiber
        - worcs_ws_empty
    - KimBaldry
        - BIO-MATE

Assignment: **CJ**

Done

4. Reviewer #4

Submitted on 26/Oct/2020 by Katherine Wolstencroft `https://orcid.org/0000-0002-1279-5133`

- Overall Impression: Good
- Suggested Decision: Accept
- Technical Quality of the paper: Excellent
- Presentation: Good
- Reviewer's confidence: Medium
- Significance: Moderate significance
- Background: Reasonable
- Novelty: Limited novelty
- Data availability: All used and produced data (if any) are FAIR and openly available in established data repositories
- Length of the manuscript: The length of this manuscript is about right

This paper describes a Workflow for Open Reproducible Code in Science (WORCS). It promotes reproducible and FAIR R code and complies with the TOP-guidelines. The workflow is presented with an R library and a Github repository containing worcs templates and all supporting materials for this manuscript.

Reasons to accept:

The paper is well written and clearly describes WORCS, which promotes reproducible and open code for scientific investigations. It complies with current best practices for open and reproducible code and it also follows the FAIR principles. Such initiatives help promote better open and reproducible science. The approach taken here is pragmatic and lightweight, which is necessary for large scale uptake. The authors also focus on ease of use and provide a "one-click solution".

WORCS is for scientists working with R with RMarkdown. It could therefore be argued that the utility is limited. However, R is widely used across science and the approach taken here could serve as an example for other software.

Reasons to reject:

RC: **C1: The abstract of the paper states that the manuscript "provides examples of the implementation of worcs in R", so I was expecting science projects that had been described using worcs. If these examples exist, they should be referenced and linked to from the manuscript, as they would provide a better demonstration of the practical utility of the workflow and software than tutorial style examples.**

AR: Response: We thank the Reviewer for this helpful suggestion! Although such user examples are not yet numerous, the lead author has six public WORCS repositories, and there are four public repositories by other users. In our opinion, it would be most useful to have a continuously cumulating list of example repositories, and link that list in the paper - rather than to publish a static list. Therefore, to address this comment, we have scripted a web scraper that searches GitHub for WORCS repositories (using the metadata tags created by the `worcs` package). We have embedded the scraper in the README.Rmd file on the worcs GitHub page, which is regularly updated. In the revised paper, we point readers to a list of public `worcs` projects in the Abstract, and when introducing the workflow.

Assignment: **CJ**

Done

RC: **C2: There is a comparison in the paper to another R-based reproducible science solution (Peikert and Brandmaier (2019)). The authors discuss some of the differences between these approaches, but the comparison is limited. It is not clear to me how different these approaches actually are, apart from**

ease of use. If scientists are already familiar with R, is there a significant learning curve with either solution?

AR: We agree that this is an important point. We have attemptedto address it by rewriting the paragraph comparing these two solutions (as cited below). We hope that this is satisfactory. At this point in time, this is all we can say about the comparison of the two workflows, because the workflow of Peikert and Brandmaier is still in press, and the software implementation has not yet been fully developed.

> A different class of solutions instead focuses on the practical issue of *how* researchers can meet the requirements of open science. A notable example is the workflow for reproducible analyses developed by Peikert and Brandmaier (n.d.), which uses Docker to ensure strict computational reproducibility for even the most sophisticated analyses. There are some decisive differences with WORCS. First, concerning the scope of the workflow: WORCS is designed to address a unique issue not covered by other existing solutions, namely, to provide a workflow most conducive to satisfying TOP-guidelines 1-7 while adhering to the FAIR principles. Peikert and Brandmaier instead focus primarily on computational reproducibility, which is relevant for TOP-guidelines 2, 3, and 5. Second, with regard to ease of use, WORCS aims to bring down the learning curve by adopting sensible defaults for any decisions to be made. Peikert and Brandmaier, by contrast, designed their workflow to be very flexible and comprehensive, thus requiring substantially more background knowledge and technical sophistication from users. To sum up, WORCS builds upon the same general principles as Peikert and Brandmaier, and the two workflows are compatible. What sets WORCS apart is that it is more lightweight in terms of system and user requirements, thereby facilitating adoption, but still ensures computational reproducibility under *most circumstances*.

Assignment: **CJ, AP, AB**

Done

5. Editor

Submitted by Tobias Kuhn on Tue, 10/27/2020 - 02:04, http://orcid.org/0000-0002-1267-0234

RC: The reviewers agree that the paper has merit, but some of them also point to major shortcomings, in particular the lack of discussion of user adoption and of handling non-linear research processes. Note that according to our guidelines, resource papers do not need to have a full-blown user evaluation, but we expect "sound evidence of its (potential for) reuse".

AR: Response: We thank the Editor and all Reviewers for their helpful comments on this manuscript. We have attempted to address all of these points in the action letter above. With regard to the specific point reiterated by the Editor:

5.0.1 User adoption

We thank the Editor for pointing out that a full-blown user evaluation is not required for a resource paper. Of course, we take this Reviewer suggestion very seriously, and intend to conduct a user evaluation after the paper is published and users begin to adopt the workflow. Please refer to our response to Reviewer 1's

comment 1, where we discuss our efforts to ensure the *potential for reuse*, and provide preliminary evidence for actual reuse/uptake of the workflow.

5.0.2 Non-linear research processes

Although Reviewer 3 claims that WORCS assumes a linear research process, this is incorrect.

We agree with Reviewer 3 that researchers often go back and forth in their work. We did not intend to convey the impression that the process of conducting research is assumed to be linear. Of course, since we are presenting a workflow - there are a number of steps involved, which we have placed in roughly chronological order and numbered for reference purposes. For example, Study Design must logically occur prior to Writing and Analysis, and Publication can only logically occur after Writing. We also wish to emphasize that the "tool" (we assume the Reviewer is referring to the `worcs` R-package) does not enforce any kind of linearity.

To address Reviewer 3's comment, we have made two changes: First, we now explicitly state, in two places in the paper, and in the Workflow vignette included with the `worcs` package, that the steps are not necessarily linear:

Introducing the workflow:

> Note that, although the steps are numbered for reference purposes, we acknowledge that the process of conducting research is not always linear.

Limitations:

> Another limitation is the fact that no single workflow can suit all research projects. Not all projects conform to the steps outlined here; some may skip steps, add steps, or follow similar steps in a different order. In principle, WORCS does not enforce a linear order, and allows extentensive user flexibility.

Second, while it was always possible to manually add a manuscript or preregistration at a later point in time after creating a new project in RStudio - we have added more functions to increase the non-linear flexibility of the `worcs` package: `add_manuscript()` and `add_preregistration()`.

Assignment: **AP, AB**

Done

6. References

Allaire, J. J., Kevin Ushey, RStudio, and Yuan Tang. 2020. "R Markdown Python Engine." 2020. `https://rstudio.github.io/reticulate/articles/r_markdown.html`.

Blischak, John D., Emily R. Davenport, and Greg Wilson. 2016. "A Quick Introduction to Version Control with Git and GitHub." *PLOS Computational Biology* 12 (1): e1004668. `https://doi.org/10.1371/journal.pcbi.1004668`.

Grolemund, Garrett, and Hadley Wickham. 2017. *R for Data Science*. O'Reilly. https://r4ds.had.co.nz/.

Lamprecht, Anna-Lena, Leyla Garcia, Mateusz Kuzak, Carlos Martinez, Ricardo Arcila, Eva Martin Del Pico, Victoria Dominguez Del Angel, et al. 2019. "Towards FAIR Principles for Research Software." Edited by Paul Groth. *Data Science*, November, 1–23. https://doi.org/10.3233/DS-190026.

Muenchen, Robert A. 2012. "The Popularity of Data Science Software." R4stats.com. April 25, 2012. http://r4stats.com/articles/popularity/.

Nosek, B. A., G. Alter, G. C. Banks, D. Borsboom, S. D. Bowman, S. J. Breckler, S. Buck, et al. 2015. "Promoting an Open Research Culture." *Science* 348 (6242): 1422–5. https://doi.org/10.1126/science.aab2374.

Nowok, Beata, Gillian M. Raab, and Chris Dibben. 2016. "Synthpop: Bespoke Creation of Synthetic Data in R." *Journal of Statistical Software* 74 (1, 1): 1–26. https://doi.org/10.18637/jss.v074.i11.

Patil, Prasad, Roger D. Peng, and Jeffrey T. Leek. 2019. "A Visual Tool for Defining Reproducibility and Replicability." *Nature Human Behaviour* 3 (7, 7): 650–52. https://doi.org/10.1038/s41562-019-0629-z.

Peikert, Aaron, and Andreas Markus Brandmaier. n.d. "A Reproducible Data Analysis Workflow with R Markdown, Git, Make, and Docker." Accessed January 9, 2020. https://doi.org/10.31234/osf.io/8xzqy.

Ram, Karthik. 2013. "Git Can Facilitate Greater Reproducibility and Increased Transparency in Science." *Source Code for Biology and Medicine* 8 (1): 7. https://doi.org/10.1186/1751-0473-8-7.

R Core Team. 2020. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Van Lissa, Caspar J., Aaron Peikert, and Andreas M. Brandmaier. 2020. *Worcs: Workflow for Open Reproducible Code in Science* (version 0.1.5). https://cran.r-project.org/web/packages/worcs/index.html.

Vicente-Saez, Ruben, and Clara Martinez-Fuentes. 2018. "Open Science Now: A Systematic Literature Review for an Integrated Definition." *Journal of Business Research* 88 (July): 428–36. https://doi.org/10.1016/j.jbusres.2017.12.043.

Wilkinson, Mark D., Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, et al. 2016. "The FAIR Guiding Principles for Scientific Data Management and Stewardship." *Scientific Data* 3 (1): 160018. https://doi.org/10.1038/sdata.2016.18.

Xie, Yihui, J. J. Allaire, and Garrett Grolemund. 2018. *R Markdown: The Definitive Guide*. The R Series. Chapman and Hall/CRC. https://bookdown.org/yihui/rmarkdown/.