Journal of Dynamics, Monitoring and Diagnostics, 2023, 2, 42-50
https://doi.org/10.37965/jdmd.2023.141

ISTP
RESEARCH ARTICLE

Computational Reproducibility Within Prognostics and Health Management

Tim von Hahn and Chris K. Mechefske
Department of Mechanical and Materials Engineering, Queen's University, Kingston, Canada

(Received 17 October 2022; Revised 20 January 2023; Accepted 06 February 2023; Published online 09 February 2023)

Abstract: Scientific research frequently involves the use of computational tools and methods. Providing thorough documentation, open-source code, and data – the creation of reproducible computational research (RCR) – helps others understand a researcher's work. In this study, we investigate the state of reproducible computational research, broadly, and from within the field of prognostics and health management (PHM). In a text mining survey of more than 300 articles, we show that fewer than 1% of PHM researchers make their code and data available to others. To promote the RCR further, our work also highlights several personal benefits for those engaged in the practice. Finally, we introduce an open-source software tool, called PyPHM, to assist PHM researchers in accessing and preprocessing common industrial datasets.

Keywords: computational reproducibility; open-source; prognostics and health management

I. INTRODUCTION

> *An article about computational science [...] is merely advertising of the scholarship. The actual scholarship is the complete software development environment and the complete set of instructions which generated the figures.*

The above statement, provocatively expressed by Buckheit and Donoho (Buckheit & Donoho, 1995) paraphrases the thoughts of Jon Claebout [1]. Claebout, a geophysicist, became an early advocate for reproducible computational research. Simply put, reproducible computational research (RCR) is performed when "all details of the computation — code and data — are made conveniently available to others" [2].

Claebout's advocacy, in the early 1990s, came at a time when computation was ascending as a means of conducting scientific research. Researchers were wrestling with how these new tools affected the dissemination of ideas. Today, computational research is ubiquitous across a multitude of fields. In addition, the paradigms of the internet, immense computational power, massive data, and open-source software, have enabled tremendous scientific advances. Yet, creating and encouraging reproducible computational research remains a challenge [3,4].

Prognostics and health management (PHM) "is an enabling technology used to maintain the reliable, efficient, economic and safe operation of engineering equipment, systems and structures" [5]. PHM practitioners build these technologies with the same computational tools used across the breadth of modern science. As such, the field encounters similar challenges surrounding reproducible computational research.

The work presented here explores the topic of reproducible computational research from within the context of PHM. We ask four questions:

1. What does reproducible computational research look like? In Section II we present a practical example from PHM.
2. What is the state of reproducible computational research? In Section III we look at findings from the broader scientific community and discuss our own findings from within PHM.
3. What are the benefits to conducting reproducible computational research? In Section IV we discuss the underappreciated personal benefits of conducting reproducible computational research. Namely, performing computationally reproducible research yields greater exposure of one's work, stronger career opportunities, and increased personal satisfaction.
4. What are the challenges and opportunities for reproducible computational research? In Section V we segment the challenges and opportunities into three categories – experience, motivation, and resources. We discuss them broadly and from within PHM.

In addition, in Section VI we introduce an open-source software package, called PyPHM[1], to assist PHM practitioners in accessing and understanding public domain datasets and creating reproducible data workflows. Other fields have similar software packages, and we note that there is need of one within PHM. As such, we invite others to assist in this endeavor.

Our study fills a gap in literature by highlighting the importance of reproducible computational research within the field of PHM. We strive to not only raise the awareness of RCR within PHM, but to also provide concrete tooling, through PyPHM, to assist others in their RCR endeavors.

Isaac Newton wrote to his fellow scientist, Robert Hooke, that "if I have seen further, it is by standing on the shoulders of giants." Newton then passed on his ideas through his published writings and texts. However, for

Corresponding authors. Tim von Hahn (e-mail: t.vonhahn@queensu.ca); Chris K. Mechefske (e-mail: chris.mechefske@queensu.ca).

[1]PyPHM is publicly available on GitHub: https://github.com/tvhahn/PyPHM

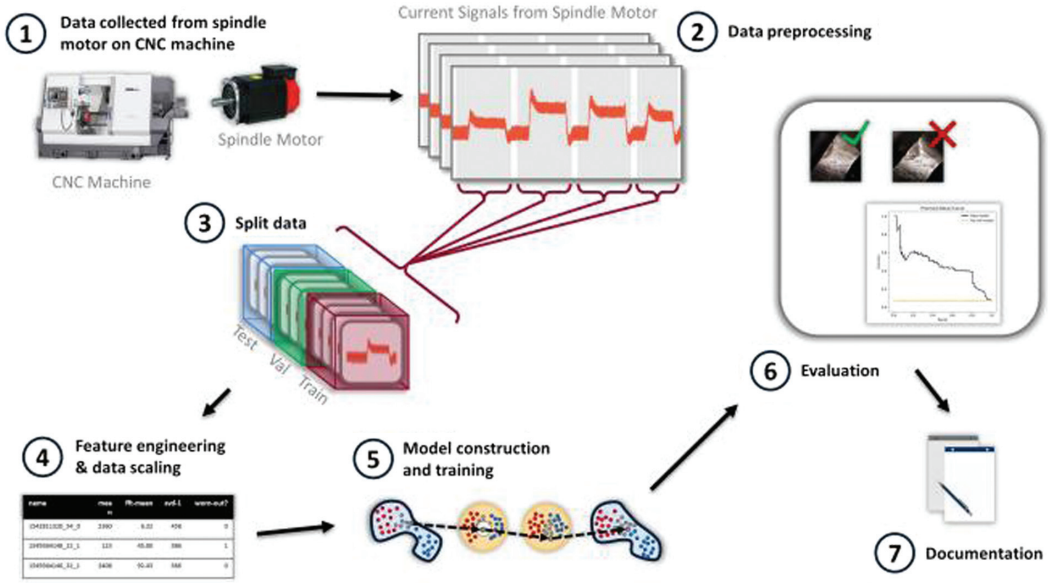

Fig. 1. The simplified steps in creating a tool wear detection and prediction model for metal machining on a CNC machine.

future generations to stand on our shoulders we should pass on more than writings. The data and the code are also needed.

II. EXAMPLE OF REPRODUCIBLE COMPUTATIONAL RESEARCH

PHM methods can be categorized into physics based and data-driven methods. Work that uses the physics of failure, to produce a PHM application, may require the code to be available if other researchers are to reproduce it. However, in a data-driven approach, both the code and data are needed to reproduce the work. The data-driven approach will be the focus in this section.

Figure 1, as an example, shows the simplified steps used to create a machine learning model for detecting tool wear on a CNC machine. The figure, combined with Table I, highlights some considerations for making the work computationally reproducible.

Table I, below, illustrates the complexities of reproducible computational research. Modern data-driven research contains intricate data preprocessing steps, feature engineering, and a complicated selection of parameters. Rarely can the details of this computational workflow be fully conveyed in a research paper. Consequently, the code, data, and additional documentation are needed to access this tacit knowledge. Unfortunately, many researchers only document their work through published papers.

III. STATE OF REPRODUCIBLE COMPUTATIONAL RESEARCH

The National Academies of Sciences, Engineering, and Medicine detailed the state of computational reproducibility in science in their 2019 report [6]. Their work showed that the lack of computational reproducibility is still a concern across science. For example, a study in computational physics demonstrated that only 6% of articles, from 307 surveyed, make the data and code available [7].

Within the AI research domain, Gunderson et al. found that fewer than 6% of articles (out of a sample of 400) provided access to their code, and fewer than 30% used a dataset that was publicly available [8]. François Chollet, the creator of Keras, a popular deep learning library, laments that many deep learning papers today are "often optimized for peer review in both style and content in ways that actively hurt clarity of explanation and reliability of results" [9]. Unfortunately, this "optimization", as expressed by Chollet, does not encourage computational reproducibility.

Does computational reproducibility fair better within PHM? Astfalck et al. surveyed 50 PHM papers between 2000 and 2014, focusing on papers building data-driven prognostic models [10]. Only eight of the papers (16%) utilized open-source or readily available datasets, and only one paper (2%) had the code available for inspection.

In the text mining process we searched for keywords and context that indicated whether the data or code, used in the research, was publicly available. As a comparison, we also text mined 100 computer science and electrical engineering articles from arXiv, an open-access archive of scholarly articles. All the articles, from across the four venues, were randomly sampled and drawn between the years of 2015 and 2021.

Figure 2 demonstrates the text mining process for a single publication venue. The process was implemented in Python using open-source libraries. The interested reader is encouraged to visit the GitHub page[2] and inspect the code of this text mining process.

In the first step, web-scraping was used to build an index of all the papers published, at a specified venue, during a certain period. The web-scraping libraries Beautiful Soup and Selenium were used. From there, as shown in step two, 100 to 150 articles were randomly selected. The pdfs of the articles were then downloaded, as shown in step three. The downloading was performed manually, or with a script, depending on the publication venue.

[2]This work is being conducted in the open and is available on GitHub: https://github.com/tvhahn/arxiv-code-search

Table I. Creating a tool wear detection and prediction model for a CNC machine and the considerations for computational reproducibility

Step	Description	Considerations for Computational Reproducibility
1	Data Collection	Electrical current data is collected from the spindle motor on a CNC machine. Operators record when the tools are changed due to wear. Considerations: • Documentation for equipment setup and collection methodology required • Meta-data for describing properties of dataset (e.g., collection frequency; time of collection; time when tools are changed; etc.) need to be provided • Data should be stored on online repository accessible to public
2	Data Preprocessing	The raw current data is broken-up into segments. Considerations: • Code and description provided showing how data is broken-up.
3	Data Splits	The data is split into training, validation, and testing data sets. Considerations: • Data splits must be done before any scaling, feature engineering, or model training • Code and description of data split methodology should be provided
4	Feature Engineering and Data Scaling	Feature engineering is conducted on the data splits. The features are then scaled. Considerations: • Code and parameters used for feature engineer and scaling are made available
5	Model Construction and Training	Model is constructed and trained on the data that has been feature engineered/scaled. Considerations: • The architecture and design decisions should be documented and made available • Parameters used in model training should be recorded • Hardware used is specified
6	Evaluation	The model performance is evaluated. Considerations: • Metrics used to evaluate are documented • Code for data visualizations is provided • Final model saved and made available to others
7	Documentation	Discussion of work is documented. Considerations: • Paper should be made available, either as open-access, or as a preprint • Code for reproducing the results should be made available on GitHub, Gitlab, or another repository • Software dependencies clearly defined

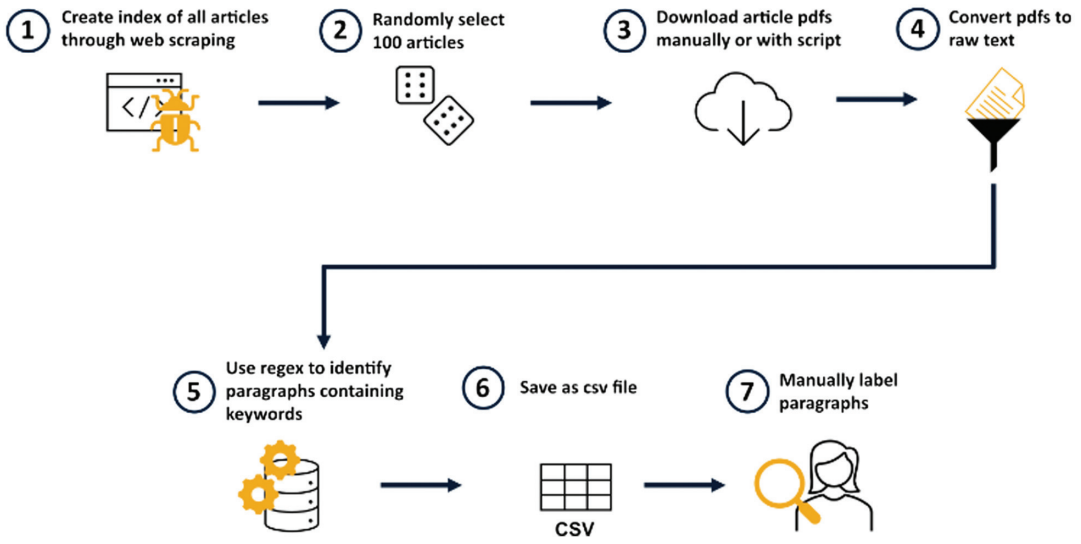

Fig. 2. The steps used in our text mining process.

In step four, the text from the pdfs was extracted using pdfminer.six. Regular expressions (regex) were implemented using Python's standard library to search for keywords and short phrases, as shown in step five. Table __ demonstrates several regular expressions, amongst many, used in the keyword search. If a keyword match was found the paragraph containing the keyword was saved into a csv file.

Finally, each paragraph in the csv file was manually labelled to indicate if the paragraph indicated publicly available data or code. The results were then aggregated across each unique article.

From the results of the text mining, as shown in Figure 3, 21% of the technical papers from the PHM Conference provided public access to their data or code. However, only 4% of the articles sampled from the PHM Conference, as shown in Figure 4, provide access to the code used in their research. In general, we observed that data is much more likely to be made publicly available than the code, regardless of the publication venue. The distributions from the other publication venues are found in Fig. A.1. in the Appendix.

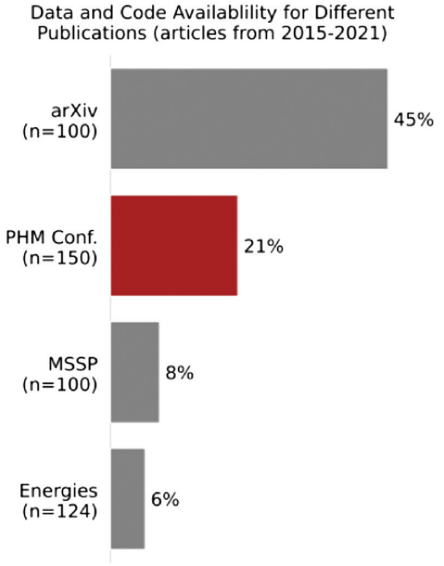

Fig. 3. Percentage of articles, from various publications, that provide public access to their data or code. The articles were randomly sampled from 2015 to 2021. The sample size (e.g. $n = 150$) for each publication venue is shown below its name.

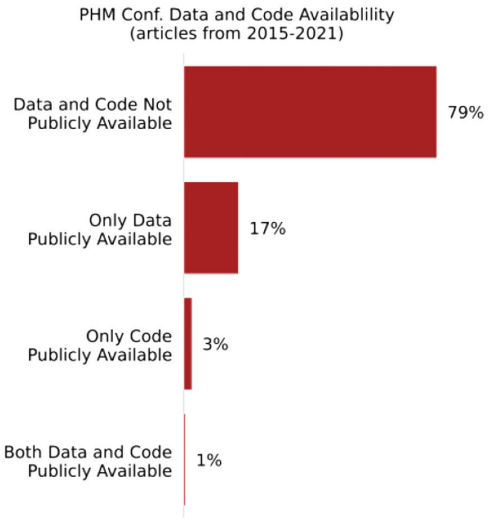

Fig. 4. The distribution of data and code availability from papers published at the PHM Conferences, between 2015 and 2021. 150 articles were randomly sampled to obtain the results.

Our text mining effort is ongoing as we seek a broader understanding of computational reproducibility within PHM and beyond, and we will further document these results in future publications. For now, we believe this observational study corroborates the evidence from Astfalck et al.; that is, the field of PHM suffers from similar issues of computational reproducibility as in other disciplines.

IV. PERSONAL BENEFITS OF REPRODUCIBLE COMPUTATIONAL RESEARCH

We observe that many commentators, when discussing computational reproducibility, appeal to the reader's sense of altruism and morality. "Reproducibility is a cornerstone of the scientific method" and therefore, it should be honored [8]. In fact, we too strongly appealed to the reader's sense of "rightness" in the introduction.

However, in this section, we highlight several benefits of computationally reproducible work that are less discussed. Rather than appealing to a sense of altruism, these appeal to the individual's self-interest. Namely, reproducible computational research can lead to increased exposure of a researcher's work, better career opportunities, and a greater sense of satisfaction.

A. INCREASED EXPOSURE

Reproducible computational research, by its nature, requires work to be open and transparent. Most often, this necessitates that the code, data, and text are made freely available on the internet. Fortunately, this additional effort does not go unrewarded.

A significant amount of research has shown that freely releasing the code, data, and published text (either through a preprint or open-access article) leads to increased citations [11–18]. In some domains, the increase was 2-fold [19]. Including the code and data, alongside scientific articles, is a clear and obvious way to produce differentiated research.

B. CAREER OPPORTUNITIES

STEM (science, technology, engineering, and mathematics) occupations are "projected to grow over two times faster than the total for all occupations in [this] decade," according to the U.S. Bureau of Labor Statistics. Computer related occupations will produce most of this growth [20]. Researchers engaged in computational science will stand to benefit as their skills are increasingly in demand.

In this competitive job market, employers have begun to accept alternate credentials, as opposed to traditional university degrees. Candidates can be hired through an intensive bootcamp program or enter a company as an apprentice. Competency can also be demonstrated through a real-world portfolio of work [21].

Creating computationally reproducible research requires the full body of work to be accessible and understandable. Therefore, not only does reproducible computational research help the scientific community, but it also creates a

strong body of work for an individual's portfolio, thus enhancing their opportunities for employment.

C. SATISFACTION

The phenomenon of open-source software (OSS) consistently raises one question: why do individuals dedicate enormous amounts of their time for little economic benefit? Intrinsic motivation – a sense of internal satisfaction – is seen as a strong driver for individuals to contribute to OSS [22,23].

The act of producing reproducible research is like that of open-source software development. One's work, through the code, data, and documents, is given to the world with no expectation of reward. Yet, the act of creating the work, and then selflessly sharing it with others, is internally satisfying. From the personal experience of the authors, this is one benefit of reproducible computational research that should not be ignored.

V. CHALLENGES AND OPPORTUNITIES

As discussed above, there are challenges in creating reproducible computational research. We have split these challenges, and subsequent opportunities, into three categories, as shown in Figure 5.

The first challenge concerns expertise. Creating reproducible computational research requires a specialized skillset that is generally covered in less depth by university curricula [6]. The skillset may include the use of version control, for both code and data; knowledge of containerization; or expertise in Linux, to name a few examples.

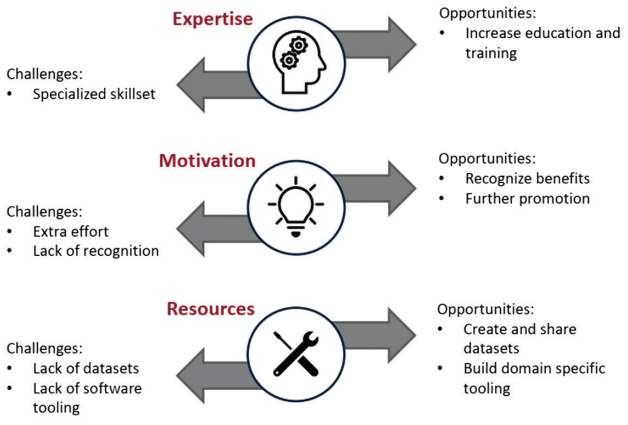

Fig. 5. The three categories of challenges and opportunities for reproducible computational research.

Education efforts are being made to improve computational researcher's expertise. Software Carpentry, a volunteer run organization, has been offering training since 1998 to improve the computational skillset of [24]. Topics of computational reproducibility have been added to the curriculum, from medicine to computer science, at multiple universities [25–27]. However, we are unaware of any courses or training specific to PHM. This is an area of opportunity.

The second challenge concerns that of motivation. Reproducible computational research requires extra effort. Unfortunately, the extra effort, combined with the pressure to publish and lack of recognition, creates an impediment to reproducibility [6]. As discussed in Section IV, we believe that a wider recognition towards the benefits of reproducible computational research can help motivate researchers. Funding organizations, and academic journals, are also encouraging researchers to consider computational reproducibility [28].

Likewise, PHM specific conferences and journals should include measures to encourage computational reproducibility. As an example of this encouragement, a measure of computational reproducibility can be integrated into the peer review process. Specific recognition can also be given to papers that demonstrate computational reproducibility. Overall, a broader discussion on how to improve computational reproducibility is warranted.

The third challenge focuses on the lack of domain specific resources, either in tooling or publicly available datasets. The computational workflow, within a specific field, may be similar across a variety of research projects. Thus, software tools can be created to standardize these workflows. The standardization allows researchers to better grasp and more quickly reproduce each other's work and avoids a myriad of ad-hoc approaches. The standardization also enables researchers to spend more time on higher-value tasks, such as developing novel algorithms, as opposed to preprocessing data. Finally, the software can be coupled with open-source datasets which facilitates the comparison of results between research groups.

Table II, below, lists several open-source software packages that are specific to certain domains. These software packages assist researchers in accessing datasets and reproducing computational workflows. The software, and their documentation, also assists in educating researchers on domain specific problems, techniques, and methods, and demonstrate how to implement solutions in a reproducible manner.

As an example of this software, consider torchvision. The software allows researchers to download common

Table II. List of well-known scientific software packages, from a variety of domains, that are used to enhance computational reproducibility

Software Name	Domain	Description
fMRIPrep [29]	Neuroimaging	Preprocessing pipeline for functional-MRI data
medicaltorch [30]	Medical imaging	General package for accessing medical imaging datasets and standardize preprocessing methods
astroML [31]	Astronomy and astrophysics	Machine learning tools and data for astronomy and astrophysics
torchvision [32]	Computer vision	Popular datasets, model architectures, and image transformations for computer vision.
Natural Language Toolkit (NLTK) [33]	Natural language processing	Open-source modules, datasets, and tutorials supporting research and development in natural language processing

Table III. A sample of keywords and phrases, along with the regex code, used in the text mining process

Keyword	Regex Code
"used dataset"	\b(used|use)(?:\W+\w+){0,5}?\W+(dataset| data set)\b
"open-source"	\b(open-source|open source|open-sourced|open sourced)\b
"code available"	\b(code)(?:\W+\w+){0,9}?\W+(available| access|download|package)

computer vision datasets, apply well recognized preprocessing techniques in a standardized way, and even load already trained deep learning models. Various tutorials and examples are available to help individuals understand the functionality of the software package.

Within PHM, there is a noted lack of high-quality, and large, datasets [34,35]. The lack of these datasets may be due to the proprietary nature of industrial data or the poor understanding of their need within the PHM research community. We encourage others to freely share their PHM datasets.

The field of PHM, to the knowledge of the authors, also lacks an open-source software package, like those found in Table III. Such a tool would enable easy access to PHM datasets and the implementation of computational workflows. We see this as an opportunity, and as such, we have begun the process of building such a software tool, discussed below.

VI. AN OPEN-SOURCE SOFTWARE TOOL FOR PHM DATASETS

Currently, PHM datasets are spread-out across the internet and require users to both find the data and then manually download it. Furthermore, users must implement their own data preprocessing steps, which can be a time-consuming process. The open-source software tool being developed, called PyPHM, will enable PHM practitioners to easily source, download, and preprocess publicly available PHM datasets in only a few lines of code. Researchers can use the preprocessed data from PyPHM, for example, for feature engineering or machine learning experiments.

Figure 6 illustrates the class hierarchy of the PyPHM software package. Specific datasets are accessible by their own class (beneath the base PyPHM class). The dataset class allows the downloading, extraction, and loading of the dataset. Finally, simple data preprocessing methods, such as windowing, are constructed in their own preprocessing method classes. PyPHM is implemented in Python and relies on common open-source libraries like NumPy, SciPy, and Pandas.

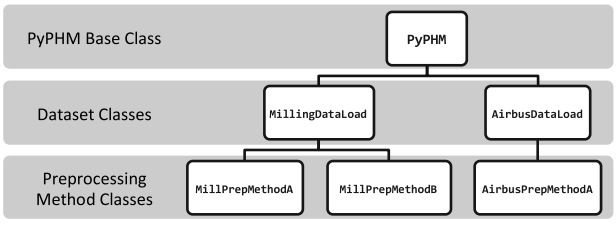

Fig. 6. Class hierarchy for the PyPHM software package. The PyPHM base class implements functionality that extends across all datasets. The dataset classes implement functionality for individual datasets (the UC-Berkeley milling dataset and the Airbus helicopter dataset are shown). Finally, simple data preprocessing methods, such as windowing, are constructed in their own preprocessing method classes. PyPHM is implemented in Python and relies on common open-source libraries like NumPy and SciPy [36,37].

Currently, there are three datasets implemented in PyPHM: the UC-Berkeley Milling Dataset, the IMS Bearing Dataset, and the Airbus Helicopter Accelerometer Dataset [38–40]. More will be implemented in the future.

PyPHM seeks to be a domain specific resource, within PHM, that can assist researchers in conducting reproducible computational research. We highlight three challenges, as noted in Section V, that PyPHM seeks to specifically address.

1. *Challenge of expertise*: PyPHM abstracts away the complexity of downloading, manipulating, and preprocessing PHM datasets. Thus, a broader audience can engage with PHM datasets, and do so in a way that can be readily reproduced by others. In addition, PyPHM is built upon common open-source tooling. Individuals can educate themselves on these tools, from a PHM context, through PyPHM's documentation and examples.
2. *Challenge of extra effort*: PyPHM allows individuals to quickly implement a standardized workflow that other researchers have used. This saves time, reduces effort, and prevents the implementation of ad-hoc workflows that are difficult for others to reproduce.
3. *Challenge due to lack of data*: Currently, PHM datasets are dispersed across the internet. PyPHM can act as an index for these PHM datasets and a central location to access them. PyPHM can also help individuals share and explore under-utilized PHM datasets once more datasets are added.

PyPHM is under active development. We welcome feedback and contributions to this nascent open-source software project. The PyPHM software package is easily accessible via the Python Package Index and on GitHub, where interested readers can also find examples and documentation to help them get started.[3]

VII. CONCLUSION AND FUTURE WORK

Computation is used across the breadth of science, and certainly within PHM. However, creating reproducible computational research remains a challenge due to the expertise required, motivational challenges, and lack of domain specific resources. Our survey of more than 300 articles, from publications engaged in PHM research, demonstrates that most researchers within PHM do not provide access to their data or code.

Despite these challenges, there are clear motivations and opportunities for improving reproducible computational research. In this work, we have highlighted three of the personal benefits of conducting reproducible computational research. Namely, creating reproducible computational research can increase a researcher's exposure, improve career opportunities, and increase one's sense of satisfaction.

[3]An example of PyPHM in use if found here: https://github.com/tvhahn/PyPHM/blob/master/notebooks/milling_example.ipynb

Furthermore, we have identified a need for an open-source software package to assist PHM researchers in accessing and preprocessing common PHM datasets. As such, we have created PyPHM, and we encourage others to assist in our efforts, either through contributions or suggestions.

Our goal for PyPHM is to continue to improve and expand its capabilities. To achieve this, we plan to increase the number of datasets that PyPHM has and refine its documentation to make it more user-friendly. Additionally, we aim to broaden the scope of our text mining survey by automating the process and incorporating advanced machine learning techniques.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflicts of interest.

References

[1] J. B. Buckheit and D. L. Donoho, "Wavelab and reproducible research," in Wavelets and Statistics: Springer, 1995, pp. 55–81.

[2] D. L. Donoho, A. Maleki, I. U. Rahman, M. Shahram, and V. Stodden, "Reproducible research in computational harmonic analysis," Computing in Science & Engineering, vol. 11, no. 1, pp. 8–18, 2008.

[3] D. C. Ince, L. Hatton, and J. Graham-Cumming, "The case for open computer programs," Nature, vol. 482, no. 7386, pp. 485–488, 2012.

[4] "Trouble at the lab," *The Economist*, Oct. 2013, [Online]. Available: https://www.economist.com/briefing/2013/10/18/trouble-at-the-lab

[5] Y. Hu, X. Miao, Y. Si, E. Pan, and E. Zio, "Prognostics and health management: a review from the perspectives of design, development and decision," Reliability Engineering & System Safety, vol. 217, p. 108063, 2022.

[6] National Academies of Sciences Engineering, Medicine, and Others, "Reproducibility and replicability in science," in National Academies Press: Washington, 2019. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/31596559/

[7] V. Stodden, M. S. Krafczyk, and A. Bhaskar, "Enabling the verification of computational results: an empirical evaluation of computational reproducibility," in *Proc. First Int. Workshop Pract. Reprod. Eval. Comput. Syst.*, Association for Computing Machinery: New York, NY, USA, 2018, pp. 1–5. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3214239.3214242

[8] O. E. Gundersen, Y. Gil, and D. W. Aha, "On reproducible AI: towards reproducible research, open science, and digital scholarship in AI publications," AI Magazine, vol. 39, no. 3, pp. 56–68, 2018.

[9] F. Chollet, Deep Learning with Python: Simon and Schuster, Shelter Island, NY, USA, 2021.

[10] L. Astfalck, M. Hodkiewicz, A. Keating, E. Cripps, and M. Pecht, "A modelling ecosystem for prognostics," Annu. Conf. PHM Soc., vol. 8, no. 1, 2016. [Online]. Available: https://papers.phmsociety.org/index.php/phmconf/article/view/2568

[11] E. Frachtenberg, "Research artifacts and citations in computer systems papers," PeerJ Comput. Sci., vol. 8, p. e887, 2022.

[12] B. F. Dorch, T. M. Drachen, and O. Ellegaard, "The data sharing advantage in astrophysics," Proc. Int. Astronom. Union, vol. 11, no. A29A, pp. 172–175, 2015.

[13] E. A. Henneken and A. Accomazzi, "Linking to data-effect on citation rates in astronomy," arXiv preprint arXiv: 1111.3618, 2011.

[14] H. A. Piwowar and T. J. Vision, "Data reuse and the open data citation advantage," PeerJ, vol. 1, p. e175, 2013.

[15] H. A. Piwowar, R. S. Day, and D. B. Fridsma, "Sharing detailed research data is associated with increased citation rate," PloS One, vol. 2, no. 3, p. e308, 2007.

[16] G. Colavizza, I. Hrynaszkiewicz, I. Staden, K. Whitaker, and B. McGillivray, "The citation advantage of linking publications to research data," PloS One, vol. 15, no. 4, p. e0230416, 2020.

[17] D. Y. Fu and J. J. Hughey, "Meta-research: releasing a preprint is associated with more attention and citations for the peer-reviewed article," Elife, vol. 8, p. e52646, 2019.

[18] G. Christensen, A. Dafoe, E. Miguel, D. A. Moore, and A. K. Rose, "A study of the impact of data sharing on article citations using journal policies as a natural experiment," PLoS One, vol. 14, no. 12, p. e0225883, 2019.

[19] A. E. Wahlquist, L. N. Muhammad, T. L. Herbert, V. Ramakrishnan, and P. J. Nietert, "Dissemination of novel biostatistics methods: impact of programming code availability and other characteristics on article citations," PloS One, vol. 13, no. 8, p. e0201590, 2018.

[20] A. Zilberman and L. Ice, "Why computer occupations are behind strong STEM employment growth in the 2019–29 decade," Computer, vol. 4, no. 5, pp. 11–15, 2021.

[21] L. Rainie and J. Anderson, "The future of jobs and jobs training.," in Pew Research Center, 2017. [Online]. Available: https://www.pewresearch.org/internet/2017/05/03/the-future-of-jobs-and-jobs-training/

[22] A. Hars and S. Ou, "Working for free?–motivations of participating in open source projects; 2001," in *34th Annu. Hawaii Int. Conf. Syst. Sci. (HICSS-34), Havaí*, 2001, pp. 25–39. [Online]. Available: https://www.computer.org/csdl/proceedings-article/hicss/2001/09817014/12OmNqI04Hm

[23] J. Bitzer, W. Schrettl, and P. J. Schröder, "Intrinsic motivation in open source software development," J. Comp. Econ., vol. 35, no. 1, pp. 160–169, 2007.

[24] G. Wilson, "Software carpentry: lessons learned," F1000 Research, vol. 3, 2014. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3976103/

[25] "An introduction to the foundations of neuro data science," McGill University. [Online]. Available: https://neurodatascience.github.io/QLS612-Overview/

[26] "Reproducible and collaborative data science," University of California, Berkeley. [Online]. Available: https://berkeley-stat159-f17.github.io/stat159-f17/

[27] "Principles, statistical and computational tools for reproducible data science," Harvard University, Oct. 2017. [Online]. Available: https://pll.harvard.edu/course/principles-statistical-and-computational-tools-reproducible-data-science

[28] V. Stodden, P. Guo, and Z. Ma, "Toward reproducible computational research: an empirical analysis of data and code policy adoption by journals," PloS One, vol. 8, no. 6, p. e67111, 2013.

[29] O. Esteban *et al.*, "fMRIPrep: a robust preprocessing pipeline for functional MRI," Nat. Methods, vol. 16, no. 1, pp. 111–116, 2019.

[30] C. S. Perone, C. Clauss, E. Saravia, P. L. Ballester, and MohitTare, "perone/medicaltorch: release v0.2," *Zenodo*, Nov. 2018. doi: 10.5281/zenodo.1495335.

[31] J. T. Vanderplas, A. J. Connolly, Ž. Ivezić, and A. Gray, "Introduction to astroML: machine learning for astrophysics," Conf. Intell. Data Understanding (CIDU), Oct. 2012, pp. 47–54. doi: 10.1109/CIDU.2012.6382200.

[32] A. Paszke *et al.*, "PyTorch: an imperative style, high-performance deep learning library," Adv. Neural Inf. Process. Syst., vol. 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[33] S. Bird, E. Klein, and E. Loper,Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit: O'Reilly Media, Inc., Sebastopol, California, United States, 2009.

[34] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, "Deep learning and its applications to machine health monitoring," Mech. Sys. Signal Process., vol. 115, pp. 213–237, 2019.

[35] W. Wang, J. Taylor, and R. J. Rees, "Recent advancement of deep learning applications to machine condition monitoring part 1: a critical review," Acoust. Australia, vol. 49, pp. 207–219, 2021.

[36] C. R. Harris *et al.*, "Array programming with NumPy," Nature, vol. 585, no. 7825, pp. 357–362, Sep. 2020. doi: 10.1038/s41586-020-2649-2.

[37] P. Virtanen *et al.*, "SciPy 1.0: fundamental algorithms for scientific computing in Python," Nat. Methods, vol. 17, pp. 261–272, 2020. doi: 10.1038/s41592-019-0686-2.

[38] A. Agogino and K. Goebel, "Milling data set. NASA Ames prognostics data repository," Moffett Field, CA, 2007, [Online]. Available: https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/

[39] J. Lee, H. Qiu, G. Yu, J. Lin, and Others, "Bearing data set," IMS, University of Cincinnati, NASA Ames Prognostics Data Repository, Rexnord Technical Services, 2007.

[40] G. R. Garcia, G. Michau, M. Ducoffe, J. S. Gupta, and O. Fink, "Temporal signals to images: monitoring the condition of industrial assets with deep learning image processing algorithms," Proc. Inst. Mech. Eng., Part O: J. Risk Reliab., vol. 236, pp. 617–627, 2021.

APPENDIX A

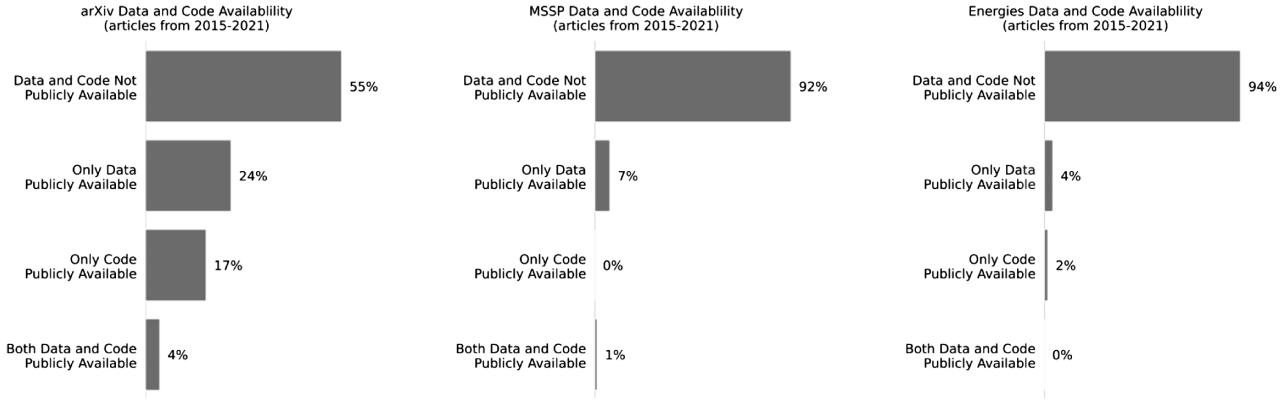

Fig. A.1. The distribution of data and code availability from papers at the arXiv, MSSP, and Energies venues. Articles were sampled between 2015 and 2021.