Davenport, JH, et al. 2020. Data Without Software Are Just Numbers. *Data Science Journal,* 19: 3, pp. 1–6. DOI: https://doi.org/10.5334/dsj-2020-003

ESSAY

# Data Without Software Are Just Numbers

James Harold Davenport[1], James Grant[2] and Catherine Mary Jones[3]

[1] Computer Science, University of Bath, Bath, UK

[2] Computing Services, University of Bath, Bath, UK

[3] Scientific Computing Department, Science and Technology Facilities Council, Oxford, UK

Corresponding author: James Harold Davenport (J.H.Davenport@bath.ac.uk)

Great strides have been made to encourage researchers to archive data created by research and provide the necessary systems to support their storage. Additionally it is recognised that data are meaningless unless their provenance is preserved, through appropriate meta-data. Alongside this is a pressing need to ensure the quality and archiving of the software that generates data, through simulation, control of experiment or data-collection and that which analyses, modifies and draws value from raw data. In order to meet the aims of reproducibility we argue that data management alone is insufficient: it must be accompanied by good software practices, the training to facilitate it and the support of stakeholders, including appropriate recognition for software as a research output.

**Keywords:** software citation; software management; reproducibility; archiving; research software engineer

## 1 Introduction

### *1.1 Context*

In the last decade there has been a drive towards improved research data management in academia, moving away from the model of 'supplementary material' that did not fit in publications, to the requirement that all data supporting research be made available at the time of publication. For instance in the UK, the Research Councils have a Concordat on Open Research Data (Research Councils UK, 2016) and the EU's Horizon 2020 programme incorporates similar policies on data availability (Horizon 2020 Programme, 2016). The FAIR principles (Wilkinson et al., 2016), Findable, Accessible, Interoperable and Re-usable embody the philosophy underlying this: Data should be preserved through archiving with a persistent identifier; it should be well described with suitable metadata; and it should be done in a way that is relevant to the domain. Together with the OpenAccess movement there has been a sea change in the availability of research and the data supporting it.

While this is a great stride *towards* transparency it does not by itself improve the quality of research, and even what exactly transparency entails remains debated (Lyon, Jeng, & Mattern, 2017). A common theme is a growing emphasis on 'reproducibility' being discussed in many disciplines (Chen et al., 2019; Mesnard & Barba, 2017; Allison, Shiffrin, & Stodden, 2018). This goes beyond 'data' and requires software and analysis pipelines to be published in a usable state alongside papers. In order to spread good practice, a coordinated effort is required: to provide training to professionalise programming in academia; to recognise the role of research software and the effort required to develop it; and to store it as well the data it creates and operates on.

In this article we discuss two cases where the use of spreadsheets highlights the need for programmatic approaches to analysis before reviewing the research software engineer movement which now has nascent organisations internationally. While some domains are adopting and at the forefront of developing good practices, the sector wide approaches needed to support their uptake generally are discussed in section 4. Finally, we summarise how data librarians and research software engineers need to work with researchers to continue to improve the situation.

## 2 When analysis 'goes wrong'

The movement towards reproducible research is driven by the belief that reviewers and readers should be able to verify and readily validate the analysis workflows supporting publications. This is not questioning academic rigour, but should be embraced as a vital part of the research cycle. Here we discuss two examples which illustrate how oversights can cause issues which should be avoidable.

### *2.1 How not to Excel … at Economics*

Reinhart and Rogoff's now notorious 2010 paper showed a headline figure of a 0.1% contraction for economies with >90% debt. (Reinhart & Rogoff, 2010) A number of issues with their work are raised by Herndon, Ash, and Pollin (2013), who were unable to reproduce the results since while the raw data was published their method was not fully described. Further, when the spreadsheet used for the calculation was analysed it was found that 5 countries (Australia, Austria, Belgium, Canada and Denmark) had been incorrectly omitted from the analysis. Together with methodological issues the revised analysis showed a 2.2% growth.

The mistakes received particular attention, with numerous articles (e.g. (Borwein & Bailey, 2013)), since the original paper was used to justify austerity policies aimed at cutting debt, in the US, UK, EU and IMF. The reliance of the proponents of these policies, and their economic and geopolitical results, on a flawed analysis, should act as a stark warning that all researchers need to mitigate against error and embrace transparency.

### *2.2 How not to Excel … with Genes*

When files are opened in MS Excel, the default behaviour is to infer data types, but while this may benefit general users, it is not always helpful. For example, two gene symbols, SEPT2 and MARCH1, are converted into dates, while certain identifiers (e.g. 2310009E13) are converted to floating point numbers. Although this has been known since 2004, a 2016 study by Ziemann, Eren, and El-Osta (2016) found that the issue continues to affect papers, as identified through supplementary data. Numbers have typically increased year-on-year, with 20% of papers affected on average rising to over 30% in Nature. This occurred despite the problem being sufficiently mature and pervasive that a service has been developed to identify affected spreadsheets (Mallona & Peidano, 2017).

## 3 Research Software

While we stress that non-programmatic approaches such as the use of spreadsheets do not of themselves cause errors, it does compromise the ability to test and reproduce analysis workflows. Further, the publication of software is part of a wider program of transparency and open access. (Munafò et al., 2017) However, if these relatively simple issues occur, we must find ways of identifying and avoiding all problems with data analysis, collection and operation of experiments. If it also makes deliberate attempts to obfuscate methods easier to identify and raise with authors at review then so much the better.

Increasingly, research across disciplines depends upon software, for experimental control or instrumentation, simulating models or analysis and turning numbers into figures. It is vital that bespoke software is published alongside the journal article and the data it supports. While it doesn't ensure that code is correct, it does enable the reproducibility of analysis and experimental workflows to be checked, and validated against correct or 'expected' behaviour. Making code available and employing good practice in its development should be the default, whether it be a million line community code a short analysis script.

The Research Software Engineer movement grew out of a working group of the *Software Sustainability Institute* (n.d.) (SSI) who have since been a strong supporter of the *UK Research Software Engineer Association* (n.d.) (UKRSEA) now Society of RSE. The aim has been to improve the sustainability, quality and recognition of research software, by advocating good software practice (e.g. Wilson et al., 2017) and career progression for its developers. Its work has resulted in recognition of the role by funders, fellowship schemes and growing recognition of software as a vital part of e-infrastructure. Its success has spawned sister organisations internationally in Germany, Netherlands, Scandanavia and the US.

A 2014 survey by the SSI showed that 92% of researchers used research software, and that 69% would not be able to conduct their research without it (Hettrick et al., 2014). Research software was defined as that used to generate, process or analyse results for publication. Furthermore 56% of researchers developed software of whom 21% had never received any form of software training. It is clear that software underpins modern research and that many researchers are involved in development, even if it is not their primary activity.

Programmatic approaches to analysis and plotting allow for greater transparency, deliver efficiencies for researchers in academia and with formal training, improve employability in industry. Their adoption is further motivated by the requirements of funders and journals which increasingly require, or at least encourage

(e.g. ACM, 2018) publication of software. This evolving landscape requires a rapid and connected response from researchers, data managers and research software engineers if institutions are to improve practice in a sustainable way.

## 4 Establishing Cultural Change

In spite of the vital role research software plays, it largely remains undervalued, with time spent in training or development seen as detracting from the 'real research'. The lack of recognition starts with funders' level of investment, the development and maintenance of code, and institutions and investigators. This is compounded by the UK's Research Assessment Exercises, and similar evaluations elsewhere, which have prioritised papers over all else. This results in inefficient development of new capability or introduction to new users, wasting researcher time and funder's investment. The lack of recognition also engrains bad habits, with the result that the longer researchers spend in academia, the lower their employability as software developers in industry. Three areas in particular are key to securing the change in culture to mirror what has been achieved with research data management.

### *4.1 Training*

In recent years organisations such as *Software Carpentry* (n.d.) have led the development of training material to professionalise the software skills of researchers. Material is available under Creative Commons licence and introduces programming skills and methods such as Unix, version control, introduction to programming languages and automation with make.

The need for such training is recognised in the recent Engineering and Physical Sciences Research Council (EPSRC) call for Centres for Doctoral Training (CDTs, one of the principal streams of research postgraduate funding in the UK):

> It is therefore a certainty that many of the students being trained through the CDTs will be using computational and data techniques in their projects, ... It is essential that they are given appropriate training so that they can confidently undertake such research in a manner that is *correct, reproducible and reusable* such as data curation and management ... (Engineering & Physical Sciences Research Council, 2018)

To achieve this there is a need to increase the number of training and range of courses. Introductory courses alone are not sufficient to deliver reproducible research, managing analysis workflows and paper writing (Mawdsley, Haines., & Jay, 2017). This requires additional in depth training and mentoring to develop programming skills, using version control appropriately for data management and automating testing. Indeed CarpentryCon events are focussing efforts to develop courses to address the recommendations of Jiménez et al. (2017).

### *4.2 Recognition*

One of the principal challenges to improving research software is the lack of recognition. In spite of the ubiquity of research software and its role in enabling research, there is no formal citation or credit in assessment exercises. Some developers are named on papers but often many are not, and there is no standardised approach to allow contributions to specific functionality or versions of code to be highlighted. A number of working groups have been addressing this (e.g. FORCE11 (Smith, Katz, & Niemeyer, 2016) and the group Working Towards Sustainable Software for Science: Practice and Experience (*WSSSPE*, n.d.)), producing number of papers and blogs posts (see e.g. (Jones Matthews, Gent, Griffin, & Tedds, 2016; Martone, 2014; Smith et al., 2016)) setting out a vision for what software citation could look like.

Services to support version control are plentiful while versioning and persistent identification of research software exist alongside research data management tools and services (such as (*Zenodo*, n.d.) and (*Figshare*, n.d.)) and the Digital Curation Centre (*Digital Curation Centre*, n.d.) have introduced a Software Management Plan template in collaboration with the SSI. Mechanisms are in place to support the development, publication and citation of software if the benefits are recognised by funders. Related to this is the benefit appropriate recognition can give to developing career pathways for research software engineers.

A further aspect is the role of software in delivering a 'digital' research experience. While computing has revolutionised research, the principal output, research papers, their look, and process of publication has barely changed. Some organisations are looking at how technology might deliver alternative experiences, and how the publication process itself might be modernised (e.g. (*F1000Research*, n.d.)).

### *4.3 Policy*

Funders and journals are key drivers for changes in the way in which research software is valued, and in having it recorded with reproducible workflows. The clear training requirements in EPSRC's CDT call is aligned with funder requirements, UK Research and Innovation (UKRI) and European Research Council (ERC) that research software, where possible, should be made freely available when the research that it enables is published. For a number of years Defense Advanced Research Projects Agency (DARPA) has published all of the software that it has supported in a single catalogue (*DARPA Open Catalogue*, n.d.).

Similarly the Nature Publishing Group (Nature, n.d.) have recently strengthened their requirements in respect of software to include its publication, and usability at time of submission to one of its journals. This should be done in a way which 'allows the reader to reproduce the published results'. Efficient working, and recognition should be sufficient carrots to engage researchers but if they don't then these changes in policy are the sticks. They will force institutions to develop policies to support research software, reproducibility and the transparency it delivers, as is happening in research data management.

Albeit one of semantics, an issue that does need to be addressed is one of terminology, in particular the terms replicate and reproduce can have different, indeed contradictory definitions. (Barba, 2018; Plesser, 2018) Once there is common language and understanding to complement the tools, the programmatic approach to data analysis and publication of software should be as ubiquitous as open data and open access are becoming.

## 5 Towards Research Software Management

Software Management in general is a much-studied subject, and many companies live or die by their software management in a way that comparatively few academic groups do. These companies may use large proprietary systems, but open-source solutions also exist and the availability of open Continuous Integration tools to automate testing means that the resource barriers to software management are much lower than they used to be. Additionally, containerisation offers potential for reproducibility since it allows the storage and re-use of the system environment when software was originally executed. The real barriers these days are the lack of consistent application of good practise across the board and the stop-start funding models too common in academia.

Encouraging the use of modern methods and professionalising training will improve the quality of research software as well as the employability and value of researchers to industry if they leave academia. It is also important that institutions continue to invest in the research software engineers to support this effort as is being seen across the UK and through schemes such as the EPSRC's RSE Fellowship calls (Engineering & Physical Sciences Research Council, 2017). Perhaps most importantly, reproducibility requires research software engineers and data librarians to work together with researchers rather than in isolation. A recent workshop at TU Delft (Cruz, Kurapati, & Türkyilmaz-van der Velden, 2018) provided an opportunity for this, but larger scale events are required to increase engagement, get us out of our silos and ensure that the tools, services and training are designed with and for the benefit of researchers.

## Competing Interests

The authors have no competing interests to declare.

## References

**ACM.** 2018. *Software and Data Artifacts in the ACM Digital Library*. https://www.acm.org/publications/artifacts.

**Allison, DB, Shiffrin, RM** and **Stodden, V.** 2018. Reproducibility of research: Issues and proposed remedies. *Proceedings of the National Academy of Sciences*, 115(11): 2561–2562. DOI: https://doi.org/10.1073/pnas.1802324115

**Barba, L.** 2018. *Terminologies for Reproducible Research*. https://arxiv.org/abs/1802.03311.

**Borwein, J** and **Bailey, D.** 2013. *The Reinhart-Rogoff error – or how not to Excel at economics*. http://theconversation.com/the-reinhart-rogoff-error-or-how-not-to-excel-at-economics-13646.

**Chen, X, Dallmeier-Tiessen, S, Dasler, R, Feger, S, Fokianos, P, Gonzalez, JB, Neubert, S,** et al. 2019. Open is not enough. *Nature Physics*, 15: 113–119. DOI: https://doi.org/10.1038/s41567-018-0342-2

**Cruz, M, Kurapati, S** and **Türkyilmaz-van der Velden, Y.** 2018. *Work-shop Report: Software Reproducibility – How to put it into practice?* https://openworking.wordpress.com/2018/06/26/workshop-report-software-reproducibility-how-to-put-it-into-practice/. DOI: https://doi.org/10.31219/osf.io/z48cm

***Darpa open catalogue.*** n.d. https://www.darpa.mil/opencatalog.

***Digital curation centre.*** n.d. https://www.dcc.ac.uk.

**Engineering & Physical Sciences Research Council.** 2017. *Research Software Engineer Fellowships II*. https://epsrc.ukri.org/funding/calls/research-software-engineer-fellowships-ii/.

**Engineering & Physical Sciences Research Council.** 2018. *EPSRC CDT Outline call*. https://www.epsrc.ac.uk/files/funding/calls/2018/2018cdtsoutlinescall/ (Accessed: 2019-08-12).

***F1000Research.*** n.d. https://f1000research.com/ (Accessed: 2019-08-12).

***Figshare.*** n.d. https://figshare.com (Accessed: 2019-08-12).

**Herndon, T, Ash, M** and **Pollin, R.** 2013, 12. Does high public debt consistently stifle economic growth? A critique of Reinhart and Rogoff. *Cambridge J. Economics*, 38(2): 257–279. DOI: https://doi.org/10.1093/cje/bet075

**Hettrick, S, Antonioletti, MLC, Chue Hong, N, Crouch, S, De Roure, D, Sufi, S,** et al. 2014. *UK research software survey 2014*. DOI: https://doi.org/10.5281/zenodo.14809

**Horizon 2020 Programme.** 2016. *Guidelines on FAIR Data Management in Horizon 2020*. http://ec.europa.eu/research/participants/data/ref/h2020/grantsmanual/hi/oa_pilot/h2020-hi-oa-data-mgt_en.pdf (Accessed: 2019-08-12).

**Jiménez, R, Kuzak, M, Alhamdoosh, M, Barker, M, Batut, B, Borg, M, Crouch, S,** et al. 2017. Four simple recommendations to encourage best practices in research software [version 1; peer review: 3 approved]. *F1000Research*, 6(876). DOI: https://doi.org/10.12688/f1000research.11407.1

**Jones, C, Matthews, B, Gent, I, Griffin, T** and **Tedds, J.** 2016. Persistent Identification and Citation of Software. *International Journal of Data Curation*, 11(2). DOI: https://doi.org/10.2218/ijdc.v11i2.422

**Lyon, L, Jeng, W** and **Mattern, E.** 2017. Research transparency: A preliminary study of disciplinary conceptualisation, drivers, tools and support services. *International Journal of Data Curation*, 12(1): 46–64. DOI: https://doi.org/10.2218/ijdc.v12i1.530

**Mallona, I** and **Peidano, M.** 2017. Truke, a web tool to check for and handle excel misidentified gene symbols. *BMC Genomics*, 18(242). DOI: https://doi.org/10.1186/s12864-017-3631-8

**Martone, M.** (ed.). 2014. *Data citation synthesis group: Joint declaration of data citation principles*. DOI: https://doi.org/10.25490/a97f-egyk

**Mawdsley, D, Haines, R** and **Jay, C.** 2017. *Reproducible Research is Software Engineering*. http://idinteraction.cs.manchester.ac.uk/RSE2017Talk/ReproducibleResearchIsRSE.html#/ (Accessed: 2019-08-12).

**Mesnard, O** and **Barba, LA.** 2017. Reproducible and replicable computational fluid dynamics: It's harder than you think. *Computing in Science Engineering*, 19(4): 44–55. DOI: https://doi.org/10.1109/MCSE.2017.3151254

**Munafò, M, Nosek, B, Bishop, D, Button, K, Chambers, C, du Sert, N, Ioannidis, J,** et al. 2017. A manifesto for reproducible science. *Nature Human Behaviour*, 1: 0021. DOI: https://doi.org/10.1038/s41562-016-0021

**Nature.** n.d. *Nature Availability Advice*. http://www.nature.com/authors/policies/availability.html.

**Plesser, HE.** 2018. Reproducibility vs. replicability: A brief history of a confused terminology. *Frontiers in Neuroinformatics*, 11: 76. DOI: https://doi.org/10.3389/fninf.2017.00076

**Reinhart, C** and **Rogoff, K.** 2010. Growth in a Time of Debt. *American Economic Review*, 100: 573–78. DOI: https://doi.org/10.1257/aer.100.2.573

**Research Councils UK.** 2016. *Concordat on Open Research Data*. https://www.ukri.org/files/legacy/documents/concordatonopenresearchdata-pdf/.

**Smith, AM, Katz, DS, Niemeyer, KE** and **FORCE11 Software Citation Working Group.** 2016. Software citation principles. *PeerJ Computer Science*, 2, e86. DOI: https://doi.org/10.7717/peerj-cs.86

***Software Carpentry.*** n.d. https://www.software-carpentry.org.

***Software Sustainability Institute.*** n.d. https://www.software.ac.uk.

***UK Research Software Engineer Association.*** n.d. http://rse.ac.uk/.

**Wilkinson, MD, Dumontier, M, Aalbersberg, IJ, Appleton, G, Axton, M, Baak, A, Mons, B,** et al. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific data*, 3. DOI: https://doi.org/10.1038/sdata.2016.18

**Wilson, G, Bryan, J, Cranston, K, Kitzes, J, Nederbragt, L** and **Teal, TK.** 2017. Good enough practices in scientific computing. *PLoS computational biology*, 13(6): e1005510. DOI: https://doi.org/10.1371/journal.pcbi.1005510

***WSSSPE.*** n.d. http://wssspe.researchcomputing.org.uk/proceedings/.

***Zenodo.*** n.d. https://zenodo.org.

**Ziemann, M, Eren, Y** and **El-Osta, A.** 2016. Gene name errors are widespread in the scientific literature. *Genome Biology*, 17: 1–3. DOI: https://doi.org/10.1186/s13059-016-1044-7

**How to cite this article:** Davenport, JH, Grant, J and Jones, CM. 2020. Data Without Software Are Just Numbers. *Data Science Journal,* 19: 3, pp. 1–6. DOI: https://doi.org/10.5334/dsj-2020-003

**Submitted:** 28 June 2018 **Accepted:** 27 November 2019 **Published:** 22 January 2020



]u[ *Data Science Journal* is a peer-reviewed open access journal published by Ubiquity Press.

OPEN ACCESS