Reproducibility by Other Means: Transparent Research Objects with Science-Oriented Provenance

Timothy McPhillips Lan Li Nikolaus Parulian Bertram Ludäscher
School of Information Sciences, University of Illinois at Urbana-Champaign
{tmcphill,lanl2,nnp2,ludaesch}@illinois.edu

1 ABSTRACT

2 INTRODUCTION

At the foundation of modern science is the expectation that the observations, experiments, and tests of predictions that comprise scientific research be independently verifiable by others. This requirement often is referred to as reproducibility (or replicability). Digital computing approaches make it possible to repeat exactly certain computational aspects of research to an extent that exceeds what can be achieved when observing or experimenting with natural phenomena in the physical world. It generally is expected that computational processes, the implementation of hardware and software enabling those processes, and the outputs of those processes all can be repeated exactly–at least in principle. Where computing makes up a significant fraction of scientific research, the terms reproducibility and replicability have come to associated with this radically different expectation of exact repeatability. Moreover, while this expectation of exact repeatability may be realizable in principle, in practice the complexities of real-world hardware and software make computational repeatedability very challenging to achieve in practice except in very limited cases. Currently much effort is going into expanding the fraction of scenarios in which the computational components of research can be automatically repeated exactly over ranges of time and space relevant to scientific research and discourse. These efforts are important for the research community to pursue, and for science funding agencies to support, especially because the computing industry generally does not have requirements for exact repeatability across signficant spans of time. However, it also is crucial to note that the concept of exact repeatability of the kind pursued by these efforts is qualitatively different from the concept of reproducibility (or replicability) that underlies the natural sciences. It is even more important to realize that scientific reproducibility is not simply a weaker form of computational repeatability. The relationship between repeatability and reproducibility as the terms are used here is not one of degree. In particular, achieving computational replicability does not automatically deliver scientific reproducibility. This is both bad and good news. The bad news is that it is possible to put much effort into achieving computational repeatability without delivering the kind of reproducibility that is critical for producing trustworthy science. The good news is that scientifically-meaningful reproducibility can be realized in cases (or over spans of time) where computational repeatability is either impossible or impractical given readily available technology. Reseachers in the natural sciences are comfortable with the idea that it may not be possible to exactly repeat all reported observations, procedures, and experimental results. They do not see this concession to real-world practicalities as a contradiction to their demand that science be reproducible. What the natural sciences actually do demand is that: (a) research procedures be repeatable by others in principle (b) the means of repeating the work be subject to review and evaluation (c) such review and evaluation be possible without actually repeating the work (d) it not be required to repeat the exacts taken in reported research to reproduce the results In other words, in the natural sciences it is actually considered a problem if exact repetition of the steps taken in reported research is required either to evaluate the work or to reproduce results. The reason this is good news for computation-intensive research is that it is not necessary to achieve or maintain perfect repeatability of the computational components of research in order for scientists to consider the work reproducible and therefore trustworthy. However, this also means that the standards, technologies, and computational best-practices that we develop and advocate in fact support scientific reproducibility. Pursuing and supporting exact computational repeatedly is not enough. In this paper we argue that the necessary ingredient of reproducible research most at risk of left out when supporting computational repeatability is transparency. We emphasisze that an area in which computer science has much to offer in supporting transparency is in the modeling, recording, and querying of the provenance of research artifacts. But we also point out that for provenance to support reprodudibility it must support science-oriented queries. Provenance must be able to answer questions about the science that was performed–not just the computations. Answers to these questions must enable others to evaluate the scientific quality of the work, and to learn what is necessary to reproduce the results without blindly repeating every step taken in the original work. And we suggest that Research Objects and related approaches are the ideal vehicle for storing and make provenance queryable in this way, thus supporting true scientific reproducibility.

3 TERMINOLOGY CHALLENGES

Challenges of terminology: Reproduce vs Replicate
The difference between them
Standardization proposals
History of usage in biology

Cell and molecular biologists study both replication and reproduction as natural processes. DNA replicates : high fidelity, variation not desired, ingredients indistiguishable, errors are corrected on the fly. Organisms reproduce: lower fidelity expected, variation desired, different ingredients acceptable. Cells have replisomes, complex molecular machines where DNA replication occurs, and copying errors are detected and corrected. In origin of life research a crucial debate is over 'replication first' (DNA World) or 'metabolism first' (aka reproduction first, i.e. without replicating genetic material). These terminologies with biology are well established. Biologists also have a rich and well-defined vocabulary to describe replicability

of experimental measurements and results. Commonly distinguish two kinds of experimental 'replicates': technical replicates, and biological replicates. FASEB definitions of reproducibility and replicability are consistent with the above. Footnote: provide number of FASEB members (105), number of societies and subdisciplines (29) represented by FASEB. Reality is that the terminology is well established in large branches of science already.

Need for reproduction/replication to mean different things in different fields

The relationships of corresponding concepts across fields is one of analogy, not identity. Exact repeatability is at least theoretically possible and sometimes practical under realistic assumptions when an experiment is entirely in silico and isolated from the outside world. As soon as observation of the real world is involved, exact repeatability often is impossible. Neither of the types of replicates in experimental biology are exact, although both are measures of repeatability. There often is no way to repeat exactly an experiment that involves scientific instruments, physical samples, or experimental apparatus. In contrast, it is not unreasonable to talk about exactly repeating a purely computational experiment, at least by the original researcher, on the same hardware, close in time to the original experiment. In reality, computational repeatability is not as easy as sometimes assumed (see below), but fundamentally this is a different situation than when a scientific instrument is involved or observations are made of the external world.

Our approach to terminology

Respect differences between fields of research and different expectations with regard to reproducibility. Do not expect standardization to even be meaningful (never mind politically achievable) across domains. Instead, only use the R* words in ways that make their meanings clear in context. Do not be surprised if computational sciences turn out not to be representative of science generally.

Specific implications for our contributions to the Research Objects field

Take care to define R* words precisely when expressing desiderata, describing features, or making or comparing claims about capabilities. Do not expect efforts to achieve computational repeatability alone to enable "reproducible science" generally.

4 COMPUTATIONAL CHALLENGES

The fundamental limitations computers impose on replicability of program executions are well known.

Finite precision arithmetic, different word sizes on different processors, round-off errors, etc, impose limits on scientific computations and their replicability across different computing environments. Virtual machines and containers do not address these issues. Full emulation is required to run the same binary in identical fashion on a different processor. This is typically slow. These limitations are even more challenging to manage reproducibly because programs typically are compiled, meaning that the exact sequence of machine instructions executed even by a single processor cannot generally be controlled. A different compiler, or a newer version of the same compiler will yield a different sequence of machine instructions.

Replicating the outputs of a program is far from straightforward

Observing that a program or set of programs can be executed again is not sufficient to conclude that the underlying computation was replicated. The outputs of the programs must be checked for equivalence. Because of the expected variation in run time behavior of programs due to the issues above, checking that outputs of a program run are equivalent to the outputs of a run of that program is not always as simple as comparing the outputs for bitwise identity. Robustly checking for equivalence of output generally must be confirmed in some way other than comparing files at the bit level. Footnote: The excellent practice of including accurate provenance and other meaningful metadata in data file headers makes it even more unlikley that outputs from different runs will be bitwise-identical.

Replicating just the software running the program is challenging in practice

How can we ensure that the stream of instructions sent to the processor for two executions is identical? Even holding the computer hardware and compiler version constant, programs depend on language libraries, OS libraries, and system calls. Much scientific software also depends directly and indirectly on large numbers of 3rd-party libraries. Only direct dependencies can be controlled reliably at build-time. And many dependencies are via shared libraries that can change between executions of the exact same executable– no recompile is needed to get a new effective executable. Footnote: Fans of the Go programming language are bringing back the static executable. Recompiling or even just rerunning the "same" program a week later can result in a completely different effective instruction stream.

Even reproducing computing environments is hard

Containers and their discontents Footnote: By 'discontents' we do not mean that we object to the use of containers, but rather than we are not content with container technology alone. There currently is much enthusiasm around containers as a means of reproducing computing environments and making computational science replicable. Whole Tale is one of several projects leveraging the capabilities of containers for this purpose. Others include Binder and Code Ocean. In Whole Tale it is recognized that containers alone cannot satisfy researcher's needs for sharing their computing environments and computations. Rather, container technology such as Docker provide an invaluable tool for the reproducible science software stack architect. A major motivation for funding (and continuing to fund) projects like Whole Tale is that the containers on their own are insufficient as means to making computational science reproducible, and it is not practical for individual researchers and groups to use containers and other technology to actually achieve scientifically meaningful reproducibility over periods longer than the publication-cycle time scale.

What containers do not do Despite what sound like suggestions to the contrary in the literature, container technology such as Docker do not ensure computational replicability, and do not on their own solve any of the problems of computational replicability described above. What containers do provide a very convenient means for executing customized computing environments on behalf of researchers, withot having to run an entire virtual machine for each environment. In common with virtual machine technology, containers do not abstract or hide the underlying hardware architecture of the computer on which they run. They do not abstract

the underlying operating system, but simply use the Linux kernel on the host. Kernel parameter settings on the host apply to all containers running on the host (reference famous blog post on the topic "Containers Do Not Contain"). Rebuilding an image from its Dockerfile specification is not guaranteed to yield the same image. It general it will not. Container images, once they are built, are not guaranteed to run on future releases of the container host. They also do not ensure that computations run within the container will replicable in the feature.

What containers are for What containers are good for is precisely what they were to do for the computing industry: enable developers to write and test code in a computing environment of a developer's choosing that can then be replicated on a very short time scale (hours or days) in staging and production environments. Containers also are good at managing conflicts in dependencies between different components of a multiprocess software architecture. Using containers to 'contain' dependencies in this way is most effective when an application can be split across multiple containers running in concert. The model of one container, one computing environment does not lend itself to dependency isolation.

The problem of time and dependencies An emerging threat to reproducibility of computational science is the spread misconception that sharing the definition of a container image, e.g. by including a Dockerfile in the Git repo for the project, is a guarantee that others (or even the original researcher) will be able to recreate the corresponding image and computing environment it represents. Researchers making this assumption may be less likely to preserve all of the information actually required to reproduce their computations. A major reason a Dockerfile is not enough is that the implicit dependencies of the built environment are constantly changing. This is well known to anyone working directly with Docker, or other container technologies. Here we will give a single example of the implications of this issue for reproducible science.

What is reproducibility really for?

Achieving meaningful replicability even for the computational parts of research is very challenging. But this is no reason to give up hope. Replicability is a means to an end–justification of scientific results–and Research Objects can help us achieve that end by other means. What is most exciting about Research Objects is that they can achieve this end despite the difficulty of computational replicability. And the primary means by which Research Objects can due this is by providing transparency.

REFERENCES