DOI: 10.14529/jsfi200106

Building a Vision for Reproducibility in the Cyberinfrastructure Ecosystem: Leveraging Community Efforts

Dylan Chapp[1], *Victoria Stodden*[2], *Michela Taufer*[1]

The scientific computing community has long taken a leadership role in understanding and assessing the relationship of reproducibility to cyberinfrastructure, ensuring that computational results – such as those from simulations – are "reproducible", that is, the same results are obtained when one re-uses the same input data, methods, software and analysis conditions. Starting almost a decade ago, the community has regularly published and advocated for advances in this area. In this article we trace this thinking and relate it to current national efforts, including the 2019 National Academies of Science, Engineering, and Medicine report on "Reproducibility and Replication in Science".

To this end, this work considers high performance computing workflows that emphasize workflows combining traditional simulations (e.g. Molecular Dynamics simulations) with *in situ* analytics. We leverage an analysis of such workflows to (a) contextualize the 2019 National Academies of Science, Engineering, and Medicine report's recommendations in the HPC setting and (b) envision a path forward in the tradition of community driven approaches to reproducibility and the acceleration of science and discovery. The work also articulates avenues for future research at the intersection of transparency, reproducibility, and computational infrastructure that supports scientific discovery.

Keywords: reproducibility, replicability, transparency, high-performance computing, molecular dynamics, in situ analytics.

Introduction

In recent years, issues of reproducibility and replicability have come to the fore in venues as diverse as scholarly publications, numerous panels and presentations at conferences and other gatherings, and publications in the scholarly literature. These discussions and topics have engaged researchers in a diverse set of disciplinary areas such as scientific computing, the life sciences, statistics, geophysics, psychology, and more. A frequent thread in these discussions is the shortcomings in the clarity, completeness, and specificity of computational and data analysis methods in research dissemination. At the same time, journal editors and scientific societies have considered approaches to making available the code and data relied on published articles. In addition, national and international research funders have and are adopting requirements to promote transparency in research artifacts, such as data and code, that result from funded research.

Some of this activity can be rooted in examples that have been raised in the research community: the journal Nature recently reported that experiments at CERN had not shown neutrinos to be faster than light as originally reported [8]; data can be lost or unavailable and analysis algorithms in proprietary codes [46]; and a recent workshop at the ACM/IEEE Supercomputing (SC) conference discussed how parallelized simulation codes can in some cases produce unexpected nondeterminism in scientific findings [28]. Attention has recently been drawn to principles for advancing reproducibility in the computational context [41–43]. As a heuristic for

[1] University of Tennessee, Knoxville, Knoxville, TN, United States

[2] National Center for Supercomputing Applications, University of Illinois at Urbana Champaign, Champaign, IL, United States

understanding the salience of reproducibility issues, a Google Scholar search for "reproducibility" and "replicability" yields over 5,000 hits in 2019 alone, compared to 2009 with fewer than 800.

In this article, we trace the context and history of discussions and efforts regarding reproducibility in the high performance computing (HPC) context and list key efforts to improving our understanding of the costs and benefits of advancing reproducibility across the cyberinfrastructure ecosystem. We then relate these efforts to the National Academies of Science, Engineering, and Medicine 2019 report on "Reproducibility and Replication in Science" [33]. We use the framing of the report to discuss three of its recommendations regarding reproducibility and replication that are particularly actionable for research teams in HPC but whose level of abstractions may create interpretation ambiguity. To address the ambiguity, we discuss and interpret these recommendations in an exemplar case, the A4MD study [45], with the aim of enabling and advancing HPC communities in their current efforts to create reproducible, replicable, and transparent HPC ecosystems for smart cyberinfrastructures [10, 14, 20, 29, 39]. We then leverage to formalisms, PRIMAD [24] and Whole Tale's Tale [15], to apply the recommendations in the use case. We conclude with a call to extend these analysis to other use cases.

1. Reproducibility in HPC Driven Communities: Overview

1.1. Community Efforts

The notion of "really reproducible research" was introduced in 1992 [18, 19, 37] and coined in 1995 [9]. The term was intended to refer to the ability to computationally regenerate the results in a publication. Since these early days this idea has been developed and applied in many contexts [22], including policy development for journals [34] and research funding, as well as best practice and guidance development for institutions, repositories, and researchers. Many challenges to these ideas have been raised [6, 16, 32]. Most recently two of the authors participated in the development of seven guidance points for the community when stepping toward computational reproducibility [42]. Several conferences including the SC conference, the PPoPP conference, and the CGO conference, are taking steps toward to enable the integration of transparency in their paper artifacts and engaging students in the effort to promote reproducibility, replicability, and transparency. One of the authors has led the effort in the past five years to make sure that the papers accepted to the SC conference have enough information to trust their results. At SC19, for the first time, all accepted papers included an appendix with a detailed artifact description of environments and methodologies that were used for achieving the key results in the papers. In pursuing the success of the reproducibility initiative, the conference has engaged the next HPC generation through the Reproducibility Challenge in the Student Cluster Competition (SCC): a paper accepted to a past SC conference is used as source for the Reproducibility Challenge of the next SC conference. SCC is an SC program that engages 16 teams (of 6 undergraduate students each) every year who are tasked to work with a vendor to build a HPC cluster from scratch and run a set of key HPC benchmarks on it during the conference. These benchmarks now include the replication of artifacts in the selected paper on the 16 different cluster architectures, creating a unique setting for practitioners to study the impact of different hardware platforms on the performance of a single common application.

1.2. Identifying Sources of Irreproducibility

First efforts to address sources of irreproducibility tackled numerical reproducibility [11, 44]. Numerical reproducibility focuses on the relationship between system software, hardware, and the ability to return bit-wise identical output [21, 27]. In the scientific domains there is generally less concern with obtaining bit-wise identical results from one study or experiment to another, however changes in the underlying computational system can give rise to uncertainties that can affect the scientific interpretation of computational results [40]. There are several possible computational sources of irreproducibility including:

- Hardware: Many fundamental operations of a computer are inherently non-deterministic. I/O devices report interrupts at unpredictable times, affecting scheduling of processes and progress of I/O, each visible to the application at the system call layer.
- Concurrency: Current systems provide high degrees of concurrency at all levels (e.g., applications use multiple processes, multiple threads, multiple cores, and/or rely on parallel accelerators like GPUs).
- Algorithmic Randomness: Many fundamental scientific algorithms rely upon random number generators: Monte Carlo sampling algorithms, random walks, and so on.
- Application Complexity: The overall application extends beyond the application code, and includes supporting libraries and services, configuration files, the operating system, and perhaps even the configuration of the network upon which it relies. It is typical to employ more than one application in the discovery process, adding to the complexity as interactions and dependencies between applications may not be well understood. Each of the environment elements may be configured and updated independently by different parties, e.g. end user(s), system and network administrators, and automatic processes.
- Provenance Capture: Assessing and verifying the significance of a data or computationally-enabled scientific finding typically requires understanding the statistical, modeling, and calibration steps taken, including the capture and reporting of negative findings and the steps used to create visualizations and figures that present results. In addition, many applications embed state information into their output to help with debugging and general provenance, however such information may not be sufficient to assess whether results that differ bit-wise are scientifically equivalent.

Applications such as Coulomb n-body atomic system simulations, planetary orbit calculations, supernova simulations all require stringent bit-wise numerical reproducibility [4].

1.3. Formalisms and Abstractions

The community is taking a structured approach to reason about and assess reproducibility in the cyberinfrastructure context at large, beyond bit-wise reproducibility. We outline the use of two formalisms to allow the community to understand the impact of changes including costs and benefits: the PRIMAD model designed to understand changes when research is replicated [24], and the "Tale" description of reproducible published computational research [15].

The PRIMAD Model: PRIMAD is a general model intended to guide reproducibility. PRIMAD helps meet an acute need in the scientific community to ground reproducibility, yet it is inherently abstract due to its applicability across all scientific domains, leading to challenges in establishing a useful level of specificity. PRIMAD breaks reproducibility into six named components (Platform, Research objective, Implementation, Methods, Actors, and Data), each of which represents an element of a computational experiment where reproducibility can be enforced by design, or conversely where a lack of such design can allow irreproducibility to seep in and potentially corrode the overall integrity of the experiment. As a first example of a PRIMAD applicability study, two of the authors have successfully evaluated the efficacy of PRIMAD as a tool for characterizing the reproducibility of more traditional applications such as real-world computational science workflows. Specifically, we examined computational workflows used to detect gravitational waves using data from the Laser Interferometer Gravitational-Wave Observatory (LIGO) [1] and the Virgo Observatory [2]. Our findings outlined how PRIMAD can be used as a general model to guide reproducibility from publications [12].

The Whole Tale "Tale": The object defined as a "Tale" is a digital bundle of artifacts and descriptors for the dissemination and publication of computational scientific findings in the scholarly record [15]. The NSF-funded Whole Tale project is developing a computational environment designed to capture the entire computational pipeline associated with a scientific experiment and thereby enable computational reproducibility [7]. In other words, research published from the Whole Tale project is published in the Tale format, which allows researchers to create and package the code, data, and information about the workflow and computational environment necessary to support, review, and recreate the computational results reported in published research. As shown in Tab. 1, the Tale captures the artifacts and information needed to facilitate greater understanding, transparency, and executability of the Tale for review and reproducibility at the time of publication.

Table 1. A manifest of objects that comprise the Whole Tale "Tale" and whose descriptions are included as Tale metadata. Adapted from [15]

Metadata	Description
Authors	List of Tale authors
Creators	Tale Creators (may differ from authors)
Title	Title of the Tale
Description	Description of the Tale
Categories	List of subject categories (keywords)
Illustration	Illustration for the Whole Tale browse page
Create Date	Date the Tale was created
Update Date	Date the Tale was last updated
License	License selected by the user
Environment	Computational environment information
Workspace	Code/scripts, workflow, narrative, documentation, data, results
External data	Data by reference to external source
Identifier	Persistent identifier for published Tale

Without standardization, decisions about what constitutes "relevant information" are inevitably ad-hoc, and may not be uniform from publication to publication or across multiple workflows within a single publication. Thus, formalisms such as PRIMAD and the Tale offer an abstraction with which to build sustainable reproducibility in a uniform fashion across scientific domains.

2. The 2019 National Academies Report Recommendations and the HPC Ecosystem

The 2019 National Academies of Science, Engineering, and Medicine (NASEM) consensus report "Reproducibility and Replication in Science", of which one of us was a committee member, found its origin in the 2017 "American Innovation and Competitiveness Act". In this Act Congress made a provision that directed the National Science Foundation to assess "research and data reproducibility and replicability issues in interdisciplinary research" and make "recommendations for improving rigor and transparency in scientific research". This opportunity offered a chance to understand the problem and the current state of reform efforts, and to articulate ways the National Science Foundation and others might improve reproducibility and replicability in research. The NASEM report set forth definitions of the terms "reproducibility" and "replication" and offers a number of recommendations regarding reproducibility and replication [33]. We discuss each of those aspects in turn.

Key words used in reproducibility discussions may have different interpretations or meanings in different disciplines and even in different discussions. For the purposes of the NASEM report the committee established the following definitions (reproduced without modification). We follow this convention in the current writing as it is consistent with previous efforts [38].

> **Reproducibility** is obtaining consistent results using the same input data, computational steps, methods, and code, and conditions of analysis. This definition is synonymous with "computational reproducibility", and the terms are used interchangeably in this report.
>
> **Replicability** is obtaining consistent results across studies aimed at answering the same scientific question, each of which has obtained its own data. Two studies may be considered to have replicated if they obtain consistent results given the level of uncertainty inherent in the system under study.
>
> **Generalizability**, another term frequently used in science, refers to the extent that results of a study apply in other contexts or populations that differ from the original one. A single scientific study may include elements or any combination of these concepts.

Reproducibility involves the original data and code; replicability involves carrying out new studies or experiments to ascertain consistency with previous answers to the same research question. In addition, these definitions suggest that when underlying digital artifacts are made accessible, the results should ideally be reproducible. However, a study conducted according to best practices and utilizing correct analysis may of course fail to replicate due to inherent uncertainties of other factors.

Among the recommendations regarding reproducibility and replication provided by the report, some are more actionable than others for research teams in the HPC setting. Accordingly, we prioritize discussion of recommendations that both describe or refer to potential changes to computational scientists' day-to-day engineering practices that could encourage or enable reproducibility and replicability of their research, and advocate for enhancements of computational scientists' software infrastructure, where success will positively impact computational science ranging from workstation-scale prototyping to studies run on leadership-class HPC systems.

Our empirical and experiential evaluation identified three NASEM report recommendations that are particularly suitable to be tailored for HPC workflows and HPC practitioners. These are reproduced from the report without modification:

> **RECOMMENDATION 4-1:** To help ensure the reproducibility of computational results, researchers should convey clear, specific, and complete information about any computational methods and data products that support their published results to enable other researchers to repeat the analysis, unless such information is restricted by non-public data policies. That information should include the data, study methods, and computational environment:
>
> - the input data used in the study either in extension (e.g., a text file or a binary) or in intension (e.g., a script to generate the data), as well as intermediate results and output data for steps that are non-deterministic and cannot be reproduced in principle;
> - a detailed description of the study methods (ideally in executable form) together with its computational steps and associated parameters; and
> - information about the computational environment where the study was originally executed, such as operating system, hardware architecture, and library dependencies (which are relationships described in and managed by a software dependency manager tool to mitigate problems that occur when installed software packages have dependencies on specific versions of other software packages).
>
> **RECOMMENDATION 5-1:** Researchers should, as applicable to the specific study, provide an accurate and appropriate characterization of relevant uncertainties when they report or publish their research. Researchers should thoughtfully communicate all recognized uncertainties and estimate or acknowledge other potential sources of uncertainty that bear on their results, including stochastic uncertainties and uncertainties in measurement, computation, knowledge, modeling, and methods of analysis.
>
> **RECOMMENDATION 6-3:** Funding agencies and organizations should consider investing in research and development of open-source, usable tools and infrastructure that support reproducibility for a broad range of studies across different domains in a seamless fashion. Concurrently, investments would be helpful in outreach to inform and train researchers on best practices and how to use these tools.

We refer the reader to the report for more details on these and the other recommendations [33]. In the next section we interpret these recommendations in the context of a specific HPC workflow, Analytics for Molecular Dynamics (A4MD).

3. Assessing the Impact of the 2019 NASEM Report Recommendations on an HPC Workflow: The A4MD Use Case

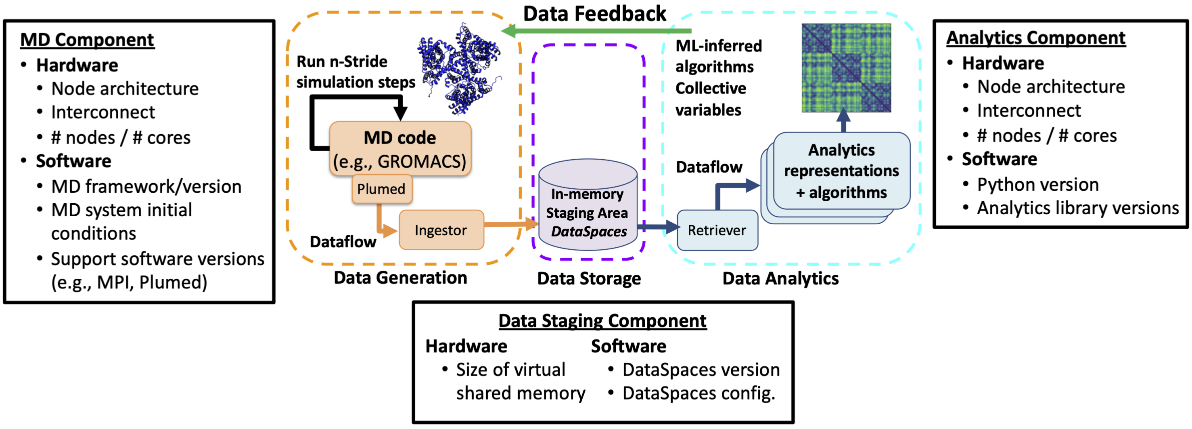
In this section we discuss the applicability of the three targeted NASEM recommendations to a real HPC use case, the Analytics for Molecular Dynamics (A4MD) workflow [45]. This use case focuses on molecular dynamics simulations that are augmented with *in situ* analytics components, thus allowing us to study a research workflow that integrates data factors into a traditionally compute intensive only project. We utilize the specific use case approach to concretize and interpret the NASEM recommendations.

3.1. The A4MD Use Case

Molecular Dynamics (MD) simulations studying the time evolution of a molecular system at atomic resolution. The fields of chemistry, material sciences, molecular biology, and drug design widely utilize MD simulations. The system sizes and time-scales accessible to MD simulations have been steadily increasing. Next-generation HPC systems will have dramatically larger compute performance than do current systems. This increase in computing capability directly translates into the ability to execute an increasing number of longer simulations and thus to expand the range of biomolecular phenomena that can be studied by MD simulation.

3.2. The NASEM Recommendations in the Context of the A4MD Workflow

The A4MD workflow presents unique challenges for compliance with the best practices outlined in NASEM recommendations, particularly in terms of capturing and disseminating the A4MD computational environment, and all of its relevant data products. Recommendation 4-1 explicitly states that the "operating system, hardware architecture, and library dependencies" of a computational experiment should be captured and shared. Since A4MD consists of three distinct computational components (the molecular dynamics simulation itself, the data staging server, and the *in situ* analytic packages), each of which may execute on separate hardware resources, the difficulty of fulfilling this requirement is magnified. We summarize in Fig. 1 a set of metadata for each of the three A4MD components that can conceivably fulfill the requirements of environment sharing specified in Recommendation 4-1. These metadata can, in principle, be captured in an automated fashion as part of the job scripts that comprise the workflow.



MD Component
- Hardware
  - Node architecture
  - Interconnect
  - # nodes / # cores
- Software
  - MD framework/version
  - MD system initial conditions
  - Support software versions (e.g., MPI, Plumed)

Analytics Component
- Hardware
  - Node architecture
  - Interconnect
  - # nodes / # cores
- Software
  - Python version
  - Analytics library versions

Data Staging Component

Hardware
- Size of virtual shared memory

Software
- DataSpaces version
- DataSpaces config.

Figure 1. Capturing A4MD's computational environment

Beyond capture of the computational environment, Recommendation 4-1 also calls for "input data used in the study either in extension (e.g., a text file or a binary) or in intension (e.g., a script to generate the data), as well as intermediate results and output data for steps that are nondeterministic and cannot be reproduced in principle". While capture of intermediate data products can potentially bolster efforts to achieve reproducibility, doing so necessarily comes at the cost of scalability, especially in the HPC setting. Efforts to achieve scalable record and replay of HPC applications indicate that capturing fine-grained data about the intermediate state of parallel executions remains an active and challenging area of research [13]. Hence, in our view the feasibility of Recommendation 4-1's guidelines regarding capture of intermediate data must be managed on a case-by-case basis. In Fig. 2 we sketch out a possible set of data products that could be recorded per execution of the A4MD workflow, ranging from the highly feasible, e.g., the input files to the MD simulation, to the highly challenging (e.g., the evolving state of the data staging server).

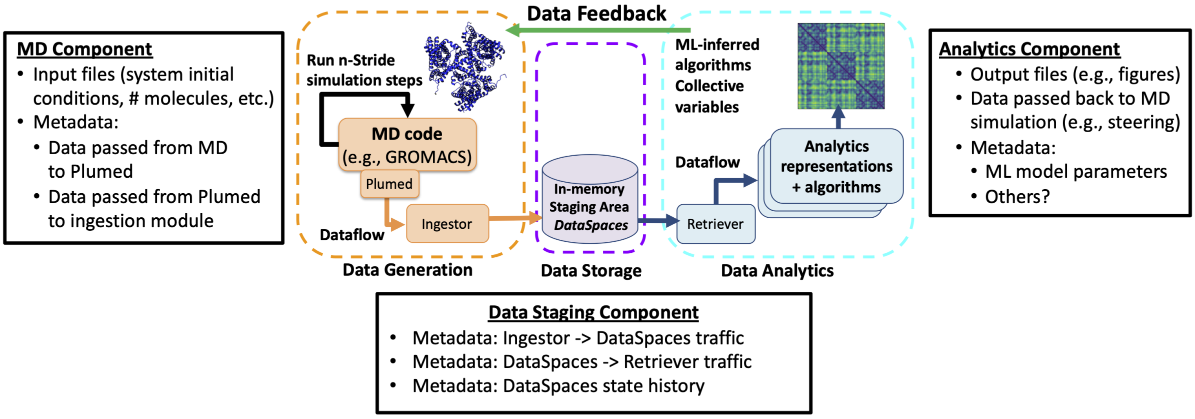

MD Component
• Input files (system initial conditions, # molecules, etc.)
• Metadata:
• Data passed from MD to Plumed
• Data passed from Plumed to ingestion module
Data Feedback
Run n-Stride simulation steps
MD code (e.g., GROMACS)
Plumed
Ingestor
Dataflow
Data Generation
In-memory Staging Area DataSpaces
Data Storage
ML-inferred algorithms Collective variables
Dataflow
Analytics representations + algorithms
Retriever
Data Analytics
Analytics Component
• Output files (e.g., figures)
• Data passed back to MD simulation (e.g., steering)
• Metadata:
• ML model parameters
• Others?
Data Staging Component
• Metadata: Ingestor -> DataSpaces traffic
• Metadata: DataSpaces -> Retriever traffic
• Metadata: DataSpaces state history

Figure 2. Capturing A4MD's input, output, and intermediary data

NASEM Recommendation 5-1 stresses the importance of uncertainty quantification in computational experiments. In an effort to comply with this recommendation, an empirical evaluation of the A4MD workflow was conducted in [45] (specifically, in Section III.C) to quantify the effect of imbalances the rate at which the MD simulation produces data and the rate at which the *in situ* analytics module consumes that data. That evaluation has provided insights for the revision of Rec. 5-1 in the next section.

Finally, in an effort to make the A4MD workflow more accessible to other researchers, the experiments described in [45] are packaged as a Jupyter notebook. However, the A4MD workflow typically involves submission of multiple interdependent jobs to a batch-scheduler; an activity not native supported by the Jupyter notebook environment. Instead, the authors of A4MD had to leverage various third-party workarounds (e.g., the signac workflow manager [3]) to encapsulate these implementation details away from the user experience of A4MD. These efforts highlight the need of the NASEM Recommendation 6-3 revision presented below, namely the need for investment in open-source, reproducibility-oriented software infrastructure.

The NASEM recommendations have broad applicability across computational settings, and we interpret the recommendations focused on in this work in the specific setting of leadership-class high performance computing (HPC) platforms. These recommendations have the highest potential for positive impact on trust and traceability of scientific findings and face unique challenges due to the characteristics of HPC platforms and HPC software.

4. The NASEM Recommendations in the HPC Setting

We leverage lessons learned from the use case in Section 3.1 to address the challenges that research teams will face when implementing a subset of the NASEM recommendations for their HPC-enabled scientific workflows. Additionally, and where appropriate, we propose HPC-tailored refinements of the recommendations with the intent of facilitating their widespread adoption. We will refer to the NASEM recommendations by their numerical identifier as given in the report (e.g., Rec. 4-1).

4.1. Recommendation 4-1: Sharing Methods, Data, and Environment

Scientific workflows, especially those deployed on HPC resources, rarely consist of a single computational element. More often, workflows consist of multiple executables that generate or consume data sets, and multiple scripts that serve functions such as pre- and post-processing of data sets, visualization, and gluing together other computational elements. Furthermore, these computational elements are not executed directly by researchers on a static resource (e.g., a single workstation), but instead are scheduled onto available resources–either by a traditional batch-scheduler (e.g., SLURM) or a more sophisticated workflow manager (e.g., Pegasus [20]).

All of these factors motivate the need to refine Rec. 4-1s request to "convey clear, specific, and complete information about any computational methods". It may be tempting to interpret this as being fulfilled by English text descriptions in publications (i.e., the text of a "Methods" or "Evaluation" section), we contend that this is insufficient–especially in the HPC setting. Instead, the "computational methods" should be described as a directed graph of executable elements. Each vertex in this graph would represent a single executable, script, or scheduled job, each associated with metadata such as commit hashes, library versions, and input parameters. In contrast with a possibly-ambiguous or incomplete English language description of "computational methods", this representation affords researchers with the ability to distinguish between structurally similar, but nevertheless distinct computational methods which is critical to investigating the reproducibility of scientific findings.

Even if an unambiguous and formal, yet shareable and ergonomic representation of "computational methods" gains traction in the scientific community, a further challenge remains: namely updating the peer-review process to appropriately evaluate study methods expressed in this form. This challenge is starting to take shape in the present day as computational notebooks (e.g., Jupyter) become more and more popular as vehicles for sharing scientific findings.

4.2. Recommendation 5-1: Broadening Notions of Uncertainty Quantification

Despite the relative ubiquity of uncertainty quantification (UQ) in the HPC setting, it is usually targeted towards probing the effect of uncertainty of inputs to simulations, rather than uncertainty inherent in the HPC platform itself. However, we contend that in the HPC setting, three factors contribute to the need to treat HPC platforms as dynamic environments in need of UQ just as much as the inputs of sensitive simulations: (1) use of multiple parallel runtimes; (2) multi-tenancy on HPC systems; and (3) opacity of code generation.

For *multiple parallel runtimes*, to cope with evolving HPC system architectures, the use of multiple parallel runtimes (e.g., MPI + OpenMP) in a single codebase has become increasingly common in scientific computing. The effect of mixing these runtimes on application-level nondeterminism has been identified as a major challenge in the push to exascale [26], and the scarcity of tools for mitigating non-determinism in these types of codebases has been documented [13].

For *multi-tenancy*, beyond the challenge of reproducing the internal state of nondeterministic applications from run to run, a greater challenge lies in reproducing the state of the system on which those applications ran, at the time that they ran. The majority of computational science on HPC systems is performed on systems in which the investigator is not the sole tenant. Thus, contention for resources such as network bandwidth between compute nodes or IO bandwidth between the system and a parallel file system can conceivably contribute to reproducibility challenges.

Finally, for *opacity of code generation*, the increasing complexity of scientific codebases coupled with the rising popularity of high-level user-friendly interfaces to them (e.g., computational notebooks) contributes to an increased risk of computational scientists being fundamentally unfamiliar with the code they execute. Incremental increases in complexity may be unavoidable for scientific codebases, we encourage computational scientists to familiarize themselves with modern tools that can increase their awareness of code-generation effects that may impede reproducibility. For example, the FLiT tool [35, 36] allows users to assess the effects of various combinations of compiler options on their codes numerical properties, while tools like Spack [25] ease the burden of maintaining and organizing multiple versions of complex scientific software built against multiple toolchains.

4.3. Recommendation 6-3: Investment in Open Source Tools to Facilitate Reproducible Research

The NASEM recommendations advocate for increased investment from funding agencies in open-source tools tailored towards reproducible research. While tools and infrastructure have emerged recently (e.g., Whole Tale [7], Popper [30, 31], ReproZip [17], Repo2Docker [23]) these tools may require that their users adhere to specific organizational patterns for their projects, or simply require additional steps in setup that researchers may find cumbersome. We contend that the fundamental tools by which researchers develop experiments ought to have reproducibility-oriented features baked in as first-class citizens [43]. In particular, we contend that computational notebooks are an attractive candidate for such an overhaul due to their increasing ubiquity; their design that co-locates data, code, and exposition; and their as-of-yet untapped capacity to capture metadata about computational experiments in support of reproducibility.

Were a funding body to invest in greenfield development of a reproducibility-oriented computation notebook environment, we suggest that the following features be prioritized: (1) automated experiment metadata collection; (2) interoperability with existing version control systems; and (3) interoperability with HPC system software.

Automated Metadata Collection: Computational notebooks present a user-friendly environment where typically, a scripting languages readevalprint loop (REPL), data visualization capabilities, and free form textual exposition, are able to be colocated. We suggest that in addition to these advantages, computational notebooks are uniquely positioned to capture metadata about computational experiments (e.g., versions of third-party libraries, identifiers for datasets, configuration details for how figures were generated) that are essential for achieving reproducibility. The HPC community has stressed the importance of collecting this metadata and provided tools for doing so, such as the SC Reproducibility Initiatives Artifact Descriptor Toolkit [5]. However the inherent drawback of tools like this is that they constitute an extra, post-hoc step for researchers–separate from their day-to-day experimental workflow. Were this functionality to be integrated directly into a reproducibility-oriented computational notebook, this metadata would be captured as a matter of course–and consequently more likely to be available to the greater scientific community.

Interoperability with version control: Currently, computational notebooks present challenges for version control. The notebook is typically stored in a hierarchical format such that small changes from the perspective of the user interface may induce relatively large changes in the underlying document (e.g., swapping cell orders in a Jupyter notebook). While this does not exclude notebooks from versioning via, e.g., Git, per se, it does render the commit history for a notebook significantly less transparent and informative than the commit history for a regular source file. A future reproducibility-oriented redesign of the computational notebook should prioritize improving integration with version control.

Interoperability with HPC system software: Despite the ease-of-use computational notebooks have enjoyed for prototyping experiments, there remain pain-points when it comes to porting these prototypes to run on large-scale HPC resources [14]. We argue that it is imperative that computational notebooks evolve to integrate seamlessly with batch schedulers so that researchers may more easily and reproducibly scale up their prototypes.

5. Applying Formalisms to Assess the NASEM Recommendations in the HPC Ecosystem

The formalisms discussed in Section 1.3 suggest guidance to understanding how to generalize findings from use cases and thereby indicate potential avenues to build sustainable reproducibility efforts in a uniform fashion across scientific domains. We identify how specific components of the two proposed formalisms (i.e., PRIMAD and the Whole Tale) can be identified or defined in order to support applicability of the three targeted recommendations (i.e., 4-1, 5-1, and 6-3) across workflows in a specific domain and for desired levels of reproducibility. The first step is to apply the formalism, the second is to update the interpretation of the three recommendations targeted in this work.

5.1. Applying the PRIMAD Formalism

We begin by presenting the elements of the PRIMAD formalism in Tab. 2. The second column of the Table applies these elements to the A4MD use case discussed in this work.

A clear division between implementation and methods in the PRIMAD model is fundamental for Rec. 4-1 but such a separation is subjective. Minor adjustments to an algorithm generally fall into implementation, yet it is hard to determine when changes are substantial enough to call it a new algorithm and thus a change in methods. In other cases, the effects of the human actors on reproducibility may be difficult to document. Even within the same research group and under consistent leadership, research objectives, and computational environments, changes in team members and shifts in member responsibility can introduce unacknowledged sources of variability. Appropriately documenting the knowledge and experience that is applied to the elements of a workflow is a challenge and it is important to understand when and how scientific results rely on specific human actions for example.

5.2. Applying the Tale Formalism

The Tale description is given in Tab. 3 with associated detail for the A4MD as best as we are able since the implementation of the A4MD use case in Whole Tale is currently underway. However, some aspects of the Tale format could be refined to fit the A4MD workflow better. In

Table 2. Applying the PRIMAD Formalism in the A4MD Use Case

PRIMAD Element	**Application to A4MD Use Case**
Platform	NERSCs Cori Cray XC40 System
Research	Molecular Dynamics (MD) simulations executed on a state-of-the-art supercomputer that characterize the impact of in situ and in transit analytics on overall MD workflow performance, and the capability for capturing rapid, rare events in the simulated molecular system.
Implementation	Two workflow configurations are run that represent in situ and in transit analytics on Haswell nodes of NERSCs Cori. Each Haswell node has two16-core Intel Xeon processors, 128GB memory, and are connected by a Cray Aries interconnect.
Methods	In the first type of workflow, because the analysis is not able to consume the frame in a timely manner, the MD either simulation waits in I/O to write to the in-memory staging area of DataSpaces (idle simulation time) or discards any frame that cannot be ingested into the staging area. In the second type of workflow the MD simulation generates a new frame with large strides and the analytics are waiting in I/O and the associated resources are idle. Trends for the time spent waiting in I/O for the simulation and idle time for the analytics are measured and observed.
Actors	Researchers at multiple institutions.
Data	MD-generated data created as output from the workflow.

particular, since the A4MD workflow consists of multiple applications (i.e., the MD simulation, the data staging server, and the *in situ* analytics modules) that potentially execute on different hardware platforms, the monolithic "environment" component of the Tale ought to be decomposed into a collection of environments. Links between various sub-components of the Tale's "workspace" and individual environments within that collection could then make explicit the correct way to set up and execute an equivalent A4MD workflow in a future replication study.

5.3. Application of the Formalisms to Our Analysis of Three NASEM Recommendations

In our analysis of the NASEM recommendations in the HPC setting, we observe significant overlap between aspects of the recommendations and the common components of reproducibility formalisms such as PRIMAD and Whole Tale. In particular, there is a clear parallel between Recommendation 4-1's emphasis on sharing study methods, computational environment, and data, and "methods", "platform", and "data" components of PRIMAD, or the "environment", "workspace", and "external data" components of Whole Tale. Our approach of leveraging formalisms helps refine and define what this might mean in particularly research settings. In Section 4.1 we discuss the potential pitfalls of compliance with Recommendation 4-1 in the HPC setting, and suggest possible refinements. Reproducibility formalisms are a natural vehicle by which those refinements can be made explicit and actionable for research teams.

Table 3. Applying the Whole Tale "Tale" Formalism in the A4MD Use Case

Tale Element	Application to A4MD Use Case
Authors	Thomas, S., Wyatt, M., Do, T.M.A., Pottier, L., da Silva, R.F., Weinstein, H.,Cuendet, M.A., Estrada, T., Deelman, E., Taufer, M.
Creators	C. Willis
Title	Characterizing in-situ and in transit analytics of molecular dynamics simulations for next generation supercomputers.
Description	This tale implements the computational pipeline associated with the publication cited in [45].
Categories	Scientific workflows, data analytics, performance, workload modeling, remote direct memory access.
Illustration	Figure 1 "Capturing A4MD's computational environment".
Create Date	February 2020
Update Date	February 2020
License	[License selected by the user]
Environment	[Computational environment information]
Workspace	Code/scripts, workflow, results
External data	None.
Identifier	As yet unpublished Tale

Elsewhere, specifically with respect to Recommendation 5-1, there is less overlap with existing reproducibility formalisms. Neither PRIMAD nor Whole Tale explicitly guide researchers towards incorporating uncertainty quantification into their studies. We contend that failure to quantify and report potential uncertainties of the computational environment can have dramatic impacts on reproducibility, and thus warrants explicit incorporation into future reproducibility formalisms. As Whole Tale is an active and ongoing development effort, there is potential to align aspects of the Tale format with Recommendation 5-1.

Finally, while in our discussion in Section 4.3 we focus on potential improvements to computational notebooks, we also see in well-funded open-source software a natural avenue for reproducibility formalisms to become useful and ubiquitous. As software tools for computational scientists mature, integration of a reproducibility formalism and tools into the common software stacks can reduce the degree of effort required for research teams to conduct reproducible experiments and disseminate sufficient information for the broader community to build on their work.

Conclusion

Even in the absence of a community-standardized formalism for reproducibility, individual research teams can nevertheless strive to comply with the NASEM recommendations, to the extent that the NASEM recommendations are sensibly interpreted for their specific use case. In this work, we presented one example of this with the A4MD workflow, and based on our example we articulated a set of refinements to Recommendations 4-1, 5-1, and 6-3 that renders them more suitable for computational science conducted in the HPC setting. We also showed an approach to making the recommendation implementations explicit and actionable through the use of a reproducibility formalism.

The results presented in this work are intended to indicate areas for further investigations. We see two principal avenues to extend our work.

First, our analysis was augmented with one single HPC use case. Still, our use case allowed us to concretize and interpret the NASEM recommendations, as well as to indicate future directions while opening the door to the empirical analysis of the impact of the recommendations in a broader HPC settings and workflows. The extension of our empirical approach based on use cases to a larger and diverse suites of HPC workflows can allow scientists and practitioners to understand the impact, costs, and benefits of the NASEM recommendations on the reproducibility of more and more complex HPC ecosystems.

Second, the two considered formalisms were not initially designed to resolve questions tackled in this work such as "how do suggested adjustments to research workflows affect HPC ecosystems as a whole and improve their reproducibility"? Still, the clarity they can each bring is an important step. Ultimately reproducibility formalisms should be further refined to completely and automatically capture the appropriate elements of the HPC ecosystem that are most impacted by the implementation of increased computational reproducibility.

Acknowledgments

This work was performed with the support of the National Science Foundation awards: OAC #1941443 EAGER: Reproducibility and Cyberinfrastructure for Computational and Data-Enabled Science; OAC #1841399 Collaborative: EAGER: Exploring and Advancing the State of the Art in Robust Science in Gravitational Wave Physics; OAC #1839010: EAGER: Preserve/Destroy Decisions for Simulation Data in Computational Physics and Beyond; and OAC #1541450: CC*DNI DIBBS: Merging Science and Cyberinfrastructure Pathways: The Whole Tale.

The authors thank Cuendet MA, and Deelman E, Estrada T, Ferreira Da Silva R, and Weinstein H, for their contributions with the A4MD workflow design and implementation.

References

1. Aasi, J., Abbott, B.P., Abbott, R., et al.: The LIGO scientific collaboration. Classical and Quantum Gravity 32(7), 074001 (2015), DOI: 10.1088/0264-9381/32/7/074001

2. Acernese, F., Agathos, M., Agatsuma, K., et al.: Advanced Virgo: a second-generation interferometric gravitational wave detector. Class.Quant. Grav. 32, 2 32(2) (2015), DOI: 10.1088/0264-9381/32/2/024001

3. Adorf, C.S., Dodd, P.M., Ramasubramani, V., et al.: Simple data and workflow management with the signac framework. Computational Materials Science 146, 220–229 (2018), DOI: 10.1016/j.commatsci.2018.01.035

4. Bailey, D., Barrio, R., Borwein, J.: High-precision computation: Mathematical physics and dynamics. Applied Mathematics and Computation 218(20), 10106–10121 (2012), DOI: 10.1016/j.amc.2012.03.087

5. Barba, L.A.: SC reproducibility initiative author-kit. `https://github.com/SC-Tech-Program/Author-Kit` (2013)

6. Barba, L.A.: The hard road to reproducibility. Science 354(6308), 142–142 (2016)

7. Brinckman, A., Chard, K., Gaffney, N., et al.: Computing environments for reproducibility: Capturing the "Whole Tale". Future Generation Comp. Syst. 94, 854–867 (2019), DOI: 10.1016/j.future.2017.12.029

8. Brumfiel, G.: Neutrinos not faster than light. ICARUS experiment contradicts controversial claim. Nature (2012)

9. Buckheit, J.B., Donoho, D.L.: WaveLab and Reproducible Research, pp. 55–81. Springer, New York, NY (1995), DOI: 10.1007/978-1-4612-2544-7_5

10. Canon, R.S., Younge, A.: A case for portability and reproducibility of HPC containers. In: IEEE/ACM International Workshop on Containers and New Orchestration Paradigms for Isolated Environments in HPC, CANOPIE-HPC, 18 Nov. 2019, Denver, CO, USA. pp. 49–54. IEEE (2019), DOI: 10.1109/CANOPIE-HPC49598.2019.00012

11. Chapp, D., Johnston, T., Taufer, M.: On the need for reproducible numerical accuracy through intelligent runtime selection of reduction algorithms at the extreme scale. In: Proceedings of the 2015 IEEE International Conference on Cluster Computing. pp. 166–175 (2015), DOI: 10.1109/CLUSTER.2015.34

12. Chapp, D., Rorabaugh, D., Brown, D.A., et al.: Applicability study of the PRIMAD model to LIGO gravitational wave search workflows. In: Proceedings of the 2nd International Workshop on Practical Reproducible Evaluation of Computer Systems, P-RECS@HPDC 2019. pp. 1–6 (2019), DOI: 10.1145/3322790.3330591

13. Chapp, D., Sato, K., Ahn, D., et al.: Record-and-replay techniques for HPC systems: A survey. Supercomputing Frontiers and Innovations 5(1), 11–30 (2018), DOI: 10.14529/jsfi180102

14. Chard, K., Gaffney, N., Hatigan, M., et al.: Toward enabling reproducibility for data-intensive research using the Whole Tale platform. In: Proceedings of the International Conference on Parallel Computing, PARCO 2019. Advances in Parallel Computing, IOS Press (2019)

15. Chard, K., Gaffney, N., Jones, M.B., et al.: Implementing computational reproducibility in the Whole Tale environment. In: Proceedings of the 2nd International Workshop on Practical Reproducible Evaluation of Computer Systems, P-RECS '19. pp. 17–22. ACM, New York, NY, USA (2019), DOI: 10.1145/3322790.3330594

16. Chen, X., Dallmeier-Tiessen, S., Dasler, R. et al.: Open is not enough. Nature Physics 15, 113–119 (2019), DOI: 10.1038/s41567-018-0342-2

17. Chirigati, F., Shasha, D., Freire, J.: Reprozip: Using provenance to support computational reproducibility. In: Presented as part of the 5th USENIX Workshop on the Theory and Practice of Provenance. USENIX (2013)

18. Claebout, J.: Hypertext documents about reproducible research (1994), `http://sepwww.stanford.edu/doku.php`

19. Claerbout, J.F., Karrenbach, M.: Electronic documents give reproducible research a new meaning. In: SEG Technical Program Expanded Abstracts 1992, pp. 601–604. Society of Exploration Geophysicists (1992), DOI: 10.1190/1.1822162

20. Deelman, E., Vahi, K., Juve, G., et al.: Pegasus: a workflow management system for science automation. Future Generation Computer Systems 46, 17–35 (2015), DOI: 10.1016/j.future.2014.10.008

21. Demmel, J., Nguyen, H.D.: Fast reproducible floating-point summation. In: 2013 IEEE 21st Symposium on Computer Arithmetic, 7-10 April 2013, Austin, TX, USA. pp. 163–172. IEEE (2013), DOI: 10.1109/ARITH.2013.9

22. Donoho, D.L., Maleki, A., Rahman, I.U., Shahram, M., Stodden, V.: Reproducible research in computational harmonic analysis. Computing in Science Engineering 11(1), 8–18 (2009), DOI: 10.1109/MCSE.2009.15

23. Forde, J., Head, T., Holdgraf, C., et al.: Reproducible research environments with repo2docker. In: ICML 2018 Reproducible Machine Learning. ICML (2018)

24. Freire, J., Fuhr, N., Rauber, A.: Reproducibility of Data-Oriented Experiments in e-Science (Dagstuhl Seminar 16041). Dagstuhl Reports 6(1), 108–159 (2016), DOI: 10.4230/DagRep.6.1.108

25. Gamblin, T., LeGendre, M., Collette, M.R., et al.: The Spack package manager: bringing order to HPC software chaos. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC'15, 15-20 Nov. 2015, Austin, TX, USA. pp. 1–12. IEEE (2015), DOI: 10.1145/2807591.2807623

26. Gopalakrishnan, G., Hovland, P.D., Iancu, C., et al.: Report of the HPC correctness summit, Jan 25–26, 2017, Washington, DC. CoRR abs/1705.07478 (2017), `https://arxiv.org/abs/1705.07478`

27. He, Y., Ding, C.H.: Using accurate arithmetics to improve numerical reproducibility and stability in parallel applications. The Journal of Supercomputing 18(3), 259–277 (2001), DOI: 10.1023/A:1008153532043

28. Honarmand, N., Torrellas, J.: Replay debugging: Leveraging record and replay for program debugging. SIGARCH Comput. Archit. News 42(3), 445–456 (2014), DOI: 10.1145/2678373.2665737

29. James, D., Wilkins-Diehr, N., Stodden, V., Colbry, D., Rosales, C., et al.: Standing together for reproducibility in large-scale computing: Report on reproducibility@xsede. CoRR abs/1412.5557 (2014), `http://arxiv.org/abs/1412.5557`

30. Jimenez, I., Arpaci-Dusseau, A., Arpaci-Dusseau, R., et al.: PopperCI: Automated reproducibility validation. In: 2017 IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS, 1-4 May 2017, Atlanta, GA, USA. pp. 450–455. IEEE (2017), DOI: 10.1109/INFCOMW.2017.8116418

31. Jimenez, I., Sevilla, M., Watkins, N., et al.: The Popper convention: Making reproducible systems evaluation practical. In: 2017 IEEE International Parallel and Distributed Processing Symposium Workshops, IPDPSW, 29 May-2 June 2017, Lake Buena Vista, FL, USA. pp. 1561–1570. IEEE (2017), DOI: 10.1109/IPDPSW.2017.157

32. Mirowski, P.: The future(s) of open science. Social Studies of Science 48(2), 171–203 (2018), DOI: 10.1177/0306312718772086

33. National Academies of Sciences, Engineering, and Medicine: Reproducibility and Replicability in Science. The National Academies Press, Washington, DC (2019), DOI: 10.17226/25303

34. Peng, R.D.: Reproducible research and biostatistics. Biostatistics 10(3), 405–408 (2009), DOI: 10.1093/biostatistics/kxp014

35. Sato, K., Laguna, I., Lee, G.L., et al.: PRUNERS: Providing reproducibility for uncovering non-deterministic errors in runs on supercomputers. The International Journal of High Performance Computing Applications 33(5), 777–783 (2019), DOI: 10.1177/1094342019834621

36. Sawaya, G., Bentley, M., Briggs, I., et al.: FLiT: Cross-platform floating-point result-consistency tester and workload. In: 2017 IEEE international symposium on workload characterization, IISWC, 1-3 Oct. 2017, Seattle, WA, USA. pp. 229–238. IEEE (2017), DOI: 10.1109/IISWC.2017.8167780

37. Schwab, M., Karrenbach, N., Claerbout, J.: Making scientific computations reproducible. Computing in Science Engineering 2(6), 61–67 (2000), DOI: 10.1109/5992.881708

38. Stodden, V.: Resolving irreproducibility in empirical and computational research. IMS Bulletin (November 2013), `http://bulletin.imstat.org/2013/11/resolving-irreproducibility-in-empirical-and-computational-research/`

39. Stodden, V., Borwein, J., Bailey, D.H.: Setting the default to reproducible in computational science research. SIAM News 46(5), 4–6 (2013), `https://sinews.siam.org/Details-Page/setting-the-default-to-reproducible-in-computational-science-research`

40. Stodden, V., Krafczyk, M.: Assessing reproducibility: An astrophysical example of computational uncertainty in the HPC context. In: The 1st Workshop on Reproducible, Customizable and Portable Workflows for HPC, HPC18 (2018)

41. Stodden, V., Leisch, F., Peng, R.D.: Implementing Reproducible Research. The R Series, Chapman & Hall/CRC (2014)

42. Stodden, V., McNutt, M., Bailey, D.H., et al.: Enhancing reproducibility for computational methods. Science 354(6317), 1240–1241 (2016), DOI: 10.1126/science.aah6168

43. Stodden, V., Miguez, S.: Best practices for computational science: Software infrastructure and environments for reproducible and extensible research. Journal of Open Research Software 2(1) (2014), DOI: 10.5334/jors.ay

44. Taufer, M., Anderson, D., Cicotti, P., et al.: Homogeneous redundancy: A technique to ensure integrity of molecular simulation results using public computing. In: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium, 4-8 April 2005, Denver, CO, USA. IEEE (2005), DOI: 10.1109/IPDPS.2005.247

45. Thomas, S., Wyatt, M., Do, T.M.A., et al.: Characterizing in situ and in transit analytics of molecular dynamics simulations for next generation supercomputers. In: Proceedings of the International Conference on eScience, eScience'19, 24-27 Sept. 2019, San Diego, CA, USA. pp. 188–198. IEEE (2019), DOI: 10.1109/eScience.2019.00027

46. Wild, S.: Irreproducible astronomy. Physics Today (2018), DOI: 10.1063/PT.6.1.20180404a