Maghami, I., Van Beusekom, A., Hay, L., Li, Z., Bennett, A., Choi, Y., Nijssen, B., Wang, S., Tarboton, D. and Goodall, J.L., 2023. Building cyberinfrastructure for the reuse and reproducibility of complex hydrologic modeling studies. Environmental Modelling & Software, 164, p.105689.

Building Cyberinfrastructure for the Reuse and Reproducibility of Complex Hydrologic Modeling Studies

Iman Maghami[a], Ashley Van Beusekom[b], Lauren Hay[b], Zhiyu Li[c], Andrew Bennett[d], YoungDon Choi[a], Bart Nijssen[b], Shaowen Wang[c], David Tarboton[e], Jonathan L. Goodall[a,*]

[a] Department of Engineering Systems & Environment, University of Virginia, Charlottesville, Virginia, USA

[b] Department of Civil & Environmental Engineering, University of Washington, Seattle, Washington

[c] Department of Geography & Geographic Information Science, University of Illinois at Urbana-Champaign, IL, USA

[d] Department of Hydrology and Atmospheric Sciences, University of Arizona, Tucson, AZ, USA

[e] Department of Civil and Environmental Engineering, Utah Water Research Laboratory, Utah State University, Logan, Utah, USA

* To whom correspondence should be addressed (E-mail: goodall@virginia.edu; Address: University of Virginia, Department of Engineering System and Environment, University of Virginia, 151 Engineers Way, P.O. Box 400747, Charlottesville, VA, 22904, USA; Tel: (434) 243-5019)

Highlights

- Presents novel cyberinfrastructure for complex hydrologic modeling studies
- Focuses on the challenges introduced by computationally and data intensive studies
- Uses Globus for large data transfers between scientific cloud services
- Leverages containerization for model portability across compute environments

- Combines model APIs and Jupyter notebooks to document modeling workflows

Abstract

Building cyberinfrastructure for the reuse and reproducibility of large-scale hydrologic modeling studies requires overcoming a number of data management and software architecture challenges. The objective of this research is to advance the cyberinfrastructure needed to overcome some of these challenges to make such computational hydrologic studies easier to reuse and reproduce. We present novel cyberinfrastructure capable of integrating HydroShare (an online data repository), CyberGIS-Jupyter for Water and high performance computing (HPC) resources (computational environments), and the Structure for Unifying Multiple Modeling Alternatives (SUMMA) hydrologic modeling framework through its application programming interface for orchestrating model runs. The cyberinfrastructure is demonstrated for a complex computational modeling study on a contiguous United States dataset. We present and discuss key capabilities of the cyberinfrastructure including 1) containerization for portability across compute environments, 2) Globus for large data transfers, 3) a Jupyter gateway to HPC environments, and 4) Jupyter notebooks for capturing the modeling workflows.

Keywords

Reproducibility; Computational Hydrology; Jupyter; HPC; Containerization

Software and Data Availability

The data and Jupyter notebooks used in this study were published on HydroShare with persistent citable Digital Object Identifiers (DOIs). A collection resource in Hydroshare (Choi et al., 2022a) holds the resources containing the data and Jupyter notebooks further described in the following table. A Hydroshare account (http://hydroshare.org) and access to the CyberGIS-Jupyter for Water computing gateway (accessed through HydroShare) are required to execute the Jupyter notebooks in the second and third resources.

Resource Description	Reference
Original NLDAS forcings for the CAMELS basins can be obtained as a NetCDF file[*]	Mizukami and Wood, 2021
SUMMA Simulations using CAMELS Datasets on CyberGIS-Jupyter for Water[**]	Choi et al., 2022b
SUMMA Simulations using CAMELS Datasets for HPC use with CyberGIS-Jupyter for Water[**]	Choi et al., 2022c

*The data from the CAMELS dataset (Newman et al., 2015a) was consolidated into one NetCDF file taking advantage of OPeNDAP data services supported by the HydroShare THREDDS server and web application connector (Tarboton and Calloway, 2021).

**The SUMMA setup for the CAMELS basins can be obtained from the summa_camels folder of the HydroShare resources.

List of relevant URLs

CyberGIS-Jupyter for Water: https://go.illinois.edu//cybergis-jupyter-water
Docker: https://www.docker.com
HydroShare REST API: https://www.hydroshare.org/hsapi/
Numpy: https://www.numpy.org
Pandas: https://pandas.pydata.org
pySUMMA: https://github.com/UW-Hydro/pysumma/releases/tag/v3.0.3
Seaborn: https://seaborn.pydata.org
Singularity: https://sylabs.io
SUMMA: https://github.com/CH-Earth/summa/releases/tag/v3.0.3
xarray: http://xarray.pydata.org
XSEDE: https://www.xsede.org

1 Introduction

Reproducibility, the ability to duplicate and verify previous findings, is a foundational principle in scientific research. In computational hydrology, Melsen et al., (2017) highlighted two contrasting definitions of model reproducibility: (1) "bit- reproducibility" which is defined as exact replication of a study, including the exact same numbers forming the results, and (2) "conclusion-reproducibility" which focuses on reproducibility of the conclusions of a study as the conclusions are expected to hold if the same experimental approach is applied. They argue that "conclusion-reproducibility" (replicating a study's conclusions) may be more important than "bit-reproducibility" (exactly replicating model runs) because hydrological theories need to be tested beyond bit-reproducibility by investigating conditions under which theories can be confirmed or falsified. Even so, conclusion-reproducibility itself goes beyond the simple sharing of code and data as open-source and online resources typically touted for achieving reproducibility. The code and data must be accompanied by well-documented workflows with readable and reusable code (Chen et al., 2020; Mullendore et al., 2021; Simmonds et al., 2022). Reusable code requires providing open-source computational environments in which the code can be executed. Ensuring this reuse and reproducibility is a non-trivial task; it requires not only adopting new capabilities for handling complex software and big data, it also requires careful software engineering practices to integrate these new capabilities into well designed and built cyberinfrastructure (Merkel, 2014).

A growing body of researchers have been discussing and proposing guidelines and strategies for reproducible computational modeling (e.g., Bush et al., 2021; Choi et al., 2021; Knoben et al., 2022; Mullendore et al., 2021; Simmonds et al., 2022). In recent work, Knoben et el. (2022) presented a novel approach for creating a hydrologic model at any location or scale (local to global) by separating model-agnostic and model-specific configuration steps within cyberinfrastructure workflows. Choi et al. (2021) described a general strategy for creating modern cyberinfrastructure to support open and reproducible hydrologic modeling as the integration of three components: (1) online data repositories; (2) computational environments leveraging containerization and self-documented computational notebooks; and (3) Application Programming Interfaces (APIs) that provide programmatic control of complex computational models. As an example of this general approach, Choi et al. (2021) also presented an implementation that used (1) HydroShare as the online repository, (2) two different Jupyter instances, one hosted by the Consortium of Universities for the Advancement of Hydrologic Science, Inc. (CUAHSI) and a second hosted by CyberGIS-Jupyter for Water, as the computational environments, and (3) pySUMMA, a Python wrapper for manipulating, running, managing, and analyzing of SUMMA (Structure for Unifying Multiple Modeling Alternatives), as the model API.

While Choi et al. (2021) focused mainly on the system design and demonstrated their approach with a fairly simple modeling use case, reproducibility in computational hydrology can present some difficult challenges when dealing with large-scale hydrologic studies (Hutton et al., 2016). These challenges mostly pertain to the use of "big data" and computationally expensive and time-consuming resources needed for reproducibility of complex hydrologic modeling studies. Hutton et al., (2016) notes that in these cases, new techniques are needed to ensure scientific rigor. In this paper, we provide an example of the overall system design outlined by Choi et al. (2021) as applied to a complex hydrologic study by Van Beusekom et al. (2022) (hereafter referred to as the VB study). We develop the necessary cyberinfrastructure to reproduce this study for selected subdomains and discuss the challenges and opportunities in ensuring conclusion-reproducibility for complex hydrologic studies.

The VB study evaluated the effect of the temporal resolution of surface meteorological inputs (or forcings) on modeled hydrological fluxes and states for 671 basins across the contiguous United States (CONUS). It quantified the difference in hydrologic outcomes based on daily or sub-daily forcings for multiple model configurations and parameter values. Reproducibility of the VB study if one was given only the input data and model code would be challenging because it requires the installation and configuration of the modeling framework SUMMA (Clark et al., 2015b, 2015a), the data volumes are very large, and the model runs require High Performance Computing (HPC) resources. The complete VB study consisted of 704 6-year model runs for each of the 671 basins (or 2.8 million model years). SUMMA was implemented with a single hydrologic response unit for each basin, resulting in a single output time series for each basin for each model configuration. For every model run, the output consisted of 14 hydrological variables, which required 6 MB per model simulation, or 2.834 TB for the entire study. While few researchers may be interested in reproducing the entire VB study, the more common use case and the focus of this study, would be to repeat or extend the VB study for a subset of the basins. We want to enable others to reproduce the VB study for subsets of the original domain as a basis for doing additional research enabling conclusion-reproducibility rather than the bit-reproducibility. For such an approach to be effective, it is not sufficient to provide the open-source SUMMA code and model input data; one must also provide the additional components described by Choi et al. (2021), i.e., computational environments, models exposed through APIs, and documented model workflows to create a cyberinfrastructure that lowers the barrier to reuse and reproducibility.

This research contributes to the growing literature advancing cyberinfrastructure for hydrology and other geoscience fields. Yang et al., (2010) illustrated the importance of using HPC in computationally intensive geospatial sciences and hydrologic modeling. Essawy et al., (2016) demonstrated server-side workflows for large-scale hydrologic data processing, although they did not make use of HPC in their application. Lyu et al., (2019) used containerization and combined computational environments including HPC and High Throughput Computing (HTC) cyberinfrastructure to directly run the models using Jupyter notebooks. Gan et al. (2020), integrated a hydrologic data and modeling web service with HydroShare as a data sharing system to show how this integration leads to a findable and reproducible modeling framework. Gichamo et al., (2020) used web-based data services to prepare input data for hydrologic models. Kurtz et al. (2017) introduced a cloud-based real-time data assimilation and modeling framework and showed how parallel processing can be used for complex hydrologic models in the cloud. However, unlike the VB study, none of Lyu et al. (2019), Gan et al. (2020), Gichamo et al. (2020) and Kurtz et al. (2017) applied their methods on a computationally extensive complex hydrologic use case. Therefore, the challenges and opportunities of using cyberinfrastructure for reproducibility of complex, large-scale hydrologic modeling. for which HPC and big data approaches are required, remain largely unexplored.

To address this research gap, we designed and implemented cyberinfrastructure to enable intuitive access to HPC computational environments and to support data transfers into and out of the HPC environment. Additionally, we provide a workflow that allows users to replicate parts of the study within their own computing environments. We also perform a workflow run-time performance analysis that compares different model scenarios by varying the size of simulations across different computing environments, providing users with a guide towards selection of the computing environment depending on the size of their simulations. The cyberinfrastructure provides a starting point for users to modify the hydrologic model setups, thus going beyond reproducibility (i.e., the ability to duplicate and verify previous findings) into replication where one modeling methodology can be used to answer the same scientific research question but with new input data (as highlighted by Essawy et al. (2020)). The cyberinfrastructure may also serve as an educational resource by providing an intuitive way for students to perform complex hydrologic modeling studies. The data and cyberinfrastructure are provided through HydroShare to run on any basin for which we provide a SUMMA setup to assist the modeler in analyzing basins individually.

The remainder of this paper is organized as follows. In Section 2, we provide a brief overview of the VB study, the cyberinfrastructure, the model workflows, and the model scenarios used for a science use case subsetted from the VB study as well as the model workflows run-time performance analysis. Section 3 provides results and discussion. The results focus on the modeling use case and an analysis of the workflow run-time performance for different computing environments. The discussion focuses on opportunities and challenges learned from our experience designing and building the cyberinfrastructure to support our modeling workflows. Finally, our conclusions and recommendations are provided in Section 4.

2 Methods

2.1 Overview of the VB study

The VB study used 671 basins to study the effects of the temporal resolution of the meteorological forcings on hydrologic model simulations across the CONUS. The basins are part of the CAMELS dataset (Catchment Attributes and MEteorology for Large-sample Studies; Newman et al., 2015b) a large-sample hydrometeorological dataset across the CONUS consisting of input forcings, basin attributes, and relevant historical streamflow records. The VB study used SUMMA (Clark et al., 2015b) to configure multiple model instances for each basin, representing eight different model configurations and 11 different sets of model parameter values. In addition, eight forcing datasets were constructed. In each of these forcing datasets one of the meteorological inputs was modified so that the diurnal cycle was replaced by the mean value over that day. The VB study performed 704 ($8\times11\times8= 704$) 6-year model runs for each CAMELS basin, consisting of one year of model initialization and five years of actual simulation. Model outputs for 14 simulated variables were stored to evaluate the sensitivity of the simulations to changes in model forcings, model configurations, and model parameters (Figure A1 and Table A1). The VB study results demonstrated that (1) the effect of each forcing input on each model output varies by model output and model location, (2) the use of a particular parameter set may not be critical in determining the most and least influential forcing variables, and (3) the choice of model physics (i.e., using different model configurations) could change the relative effect of each forcing input on model outputs.

The VB study was run with scripts on the Cheyenne supercomputer (a 5.34-petaflops, high-performance computer built for the National Center for Atmospheric Research; Computational and Information Systems Laboratory (2017)), and it took a few days to complete the runs. For each basin, the output size for a single 6-year run was 6 MB. Thus, reproducing the entire study is computationally expensive and also requires large amounts of storage (704 runs × 671 basins × 6 MB = 2.834 TB). However, the cyberinfrastructure allows individual basins to be run independently. Here, we focus on a use case in which a researcher wishes to reproduce a subset of the VB study by analyzing one or a few basins within a cloud cyberinfrastructure environment to reach conclusion-reproducibility. The conclusion-reproducibility that we aimed in this study is solely a qualitative one and if the presented cyberinfrastructure can be successfully applied to studies differing from the original study, i.e., the VB study, the conclusion- reproducibility is achieved.

2.2 Cyberinfrastructure design and implementation

Following the approach described in Choi et al. (2021), we designed and implemented cyberinfrastructure (Figure 1) to replicate the VB study by integrating (1) the HydroShare online data repository, (2) CyberGIS-Jupyter for Water Computing Gateway (CJW CG) and high-performance computational environments, and (3) a model API that can be utilized in scripts using Jupyter notebooks (here the pySUMMA API). Each of these three components is further explained in the following subsections.

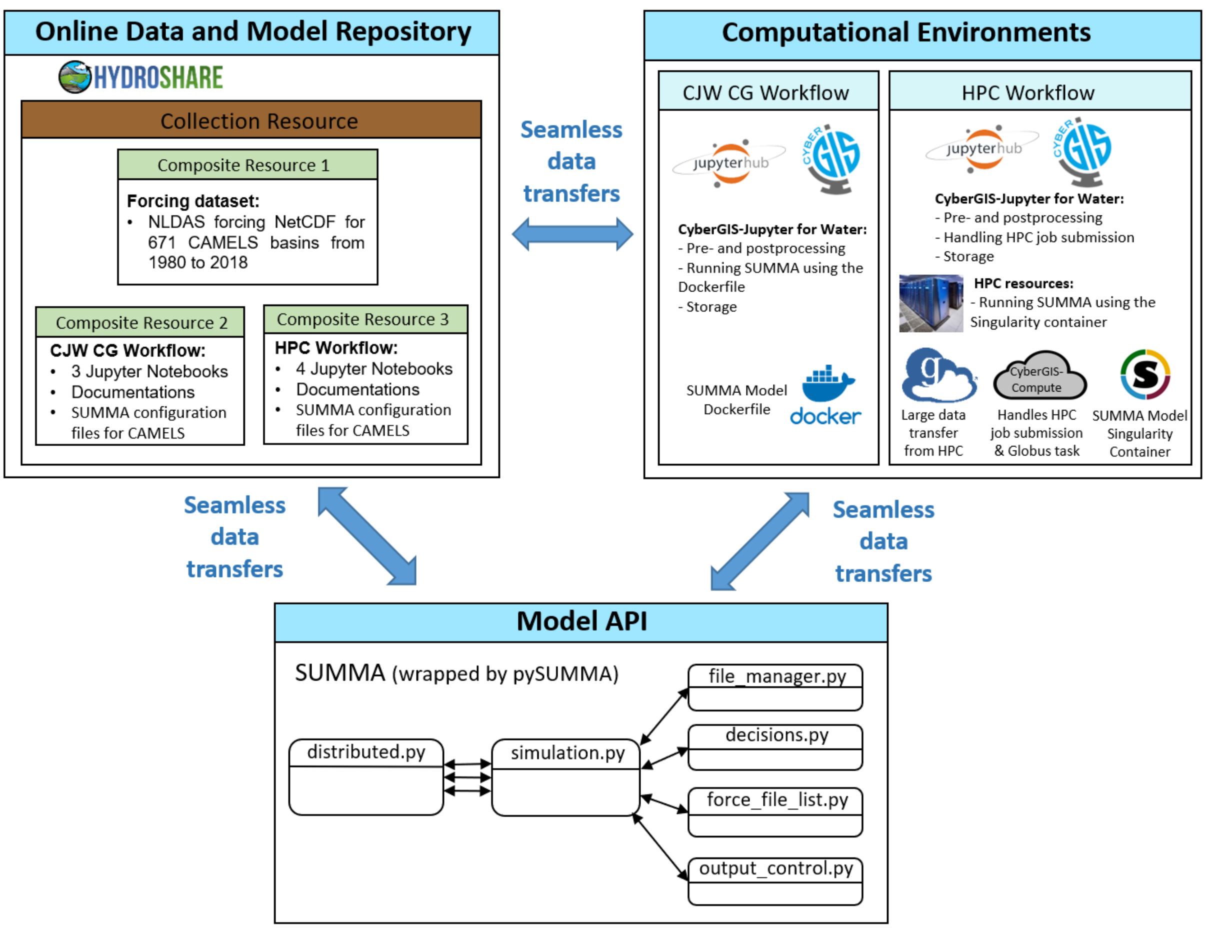

Figure 1: The three primary components of the general cyberinfrastructure (following Choi et al. 2021) with seamless data transfers for open and reproducible environmental modeling.

2.2.1 Online data repositories

We used HydroShare, an online collaboration environment, as the online data repository (Horsburgh et al., 2016; Tarboton et al., 2014). A collection resource in HydroShare, which can be found at Choi et al. (2022a), contains three resources holding the data, computational environment, and models (Figure 1).

The HydroShare resource holding the data (Mizukami and Wood, 2021) contains the forcing data set for the 671 CAMELS basins. The forcings are based on the hourly NLDAS-2 (North American Land Data Assimilation System; NLDAS-2, 2014 ; NLDAS-2 is hereafter referred to as NLDAS). The original NLDAS hourly forcing data were created on a 0.125 x 0.125 degree grid. To create hourly SUMMA model forcings, NLDAS outputs were spatially averaged over each of the 671 CAMELS basins and merged into one NetCDF file. With this format, an OPeNDAP server (OPeNDAP, 2021) can extract data for selected basins on the server, so that the user does not have to download the entire CONUS dataset to a local computer. HydroShare offers this capability via its THREDDS Data Server (TDS).

2.2.2 Computational environments

The developed computational environments provide a consistent software environment that is independent of each user's own operating system and software libraries, making it possible to study a computationally expensive research problem. Figure 2 shows each computational environments component, and the interoperability between the computational environments and HydroShare. One computational environment was implemented on the CJW CG cloud service for studies with limited computational demand, e.g., a study of only a few basins, or as an instructional tool, or for model debugging. A second computational environment was developed on an HPC resource to reproduce a problem more representative of challenges posed by the use of big-data in the VB study. The HPC environment also allows the user to study a particular basin in greater detail. In this study, the CJW CG computational environment is used to provide (1) the model execution environments configured as Docker images to enable execution of the SUMMA model for studies with limited computational demand (i.e., those need to use CJW CG Workflow), and (2) cyberinfrastructure for preprocessing, postprocessing and data storage for both studies with limited computational demand (need to use CJW CG Workflow) and with high computational demand (i.e., those need to use HPC Workflow) (Figure 2). The HPC computational environment is only used for providing model execution environments configured as Singularity containers to enable execution of the SUMMA model for studies with higher computational demand. More details on each computational environment are provided in the rest of this section.

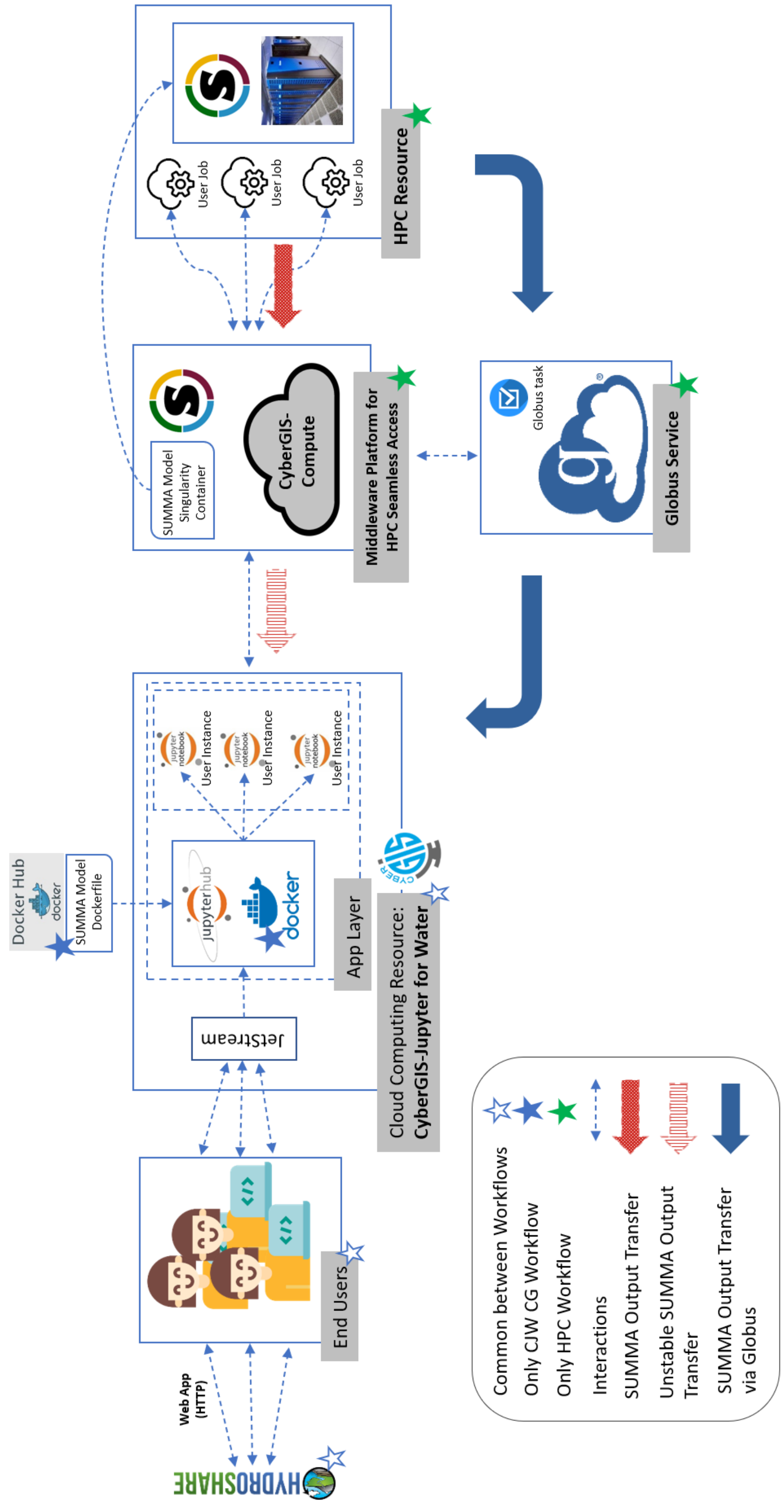

Figure 2: CJW and HPC computational environments with model execution environments configured as Docker image or Singularity container to support concurrent model execution through Jupyter notebooks, and use of Globus to transfer model outputs from HPC.

CJW CG is a cloud computing environment interoperable with HydroShare. It is an instance of CyberGISX (Yin et al., 2017) that serves the data- and computation-intensive needs of the water and environmental communities. We used CJW CG because it is publicly available, is interoperable with advanced cyberinfrastructure resources (such as the HPC resource used in this study) and has been serving the water and environmental communities to support their modeling needs.

Reproducibility was facilitated by using containerization of the SUMMA model and the pySUMMA API with Docker (Merkel, 2014) in the case of the CJW CG environment or Singularity in the case of the HPC environment (Kurtzer et al., 2017) along with a computational gateway interface to Jupyter notebooks (pySUMMA and the notebooks are described in a later section) (Figure 2). Although using Docker is a common approach to containerize the model dependencies, we used Singularity in the HPC environment because it is designed to work seamlessly with existing batch job systems to support HPC applications. The containerization and interface are hosted on the CJW scientific cloud service hosted on Jetstream cloud (Hancock et al., 2021; Stewart et al., 2015; Towns et al., 2014). The Dockerfile is hosted on a GitHub repository (Li, 2021) with pre-built docker images being shared on a Docker Hub repository. Singularity container used by the HPC environment is hosted on CyberGIS-Compute Service, a middleware platform allowing seamless access to HPC resources via Python-based Software Development Kit and core middleware services (CyberGIS-Compute Service, 2021; Li et al., 2022). The singularity container was created through docker images conversion. CyberGIS-Compute Service also handles submitting jobs to HPC as well as large data transfer from HPC through Globus (will be discussed in section 2.2.4).

The Conda software package was used to manage the project specific computational environment on CJW, allowing the user to build a Python environment with the SUMMA model, pySUMMA API, and other computational dependencies. This was done by providing a kernel version for the project (CyberGIS Center HydroShare Development Team, 2022). Using this stable kernel, which captures all the required dependencies with their specific versions, ensures careful software version control.

2.2.3 Model Application Programming Interface (API)

The model API pySUMMA was chosen to be part of the interactive tool. The pySUMMA API (Choi et al., 2021) wraps the SUMMA hydrologic modeling framework (Clark et al., 2015a) and allows the user to script the use of the SUMMA model using Python. It facilitates model configuration and allows for local execution of the model by either using a Docker container or a locally compiled SUMMA executable (Choi et al., 2021). With pySUMMA, a user can modify SUMMA input files and run SUMMA inside a Python script, as well as automatically parallelize runs and visualize output. In the simplest case the pySUMMA Simulation object wraps a single instance of a SUMMA simulation.

For users who choose to analyze multiple basins at a time in the CJW CG environment instead of the HPC environment, the notebook automatically will configure a pySUMMA Distributed object, which provides an interface to spatially distributed simulations and handles parallelism and job management under the hood. In this study, multiple SUMMA simulations are run in each basin, so a pySUMMA Ensemble object is used to manage multiple runs with different configurations. In the HPC computational environment a custom backend was written to handle parallelism using Message Passing Interface (MPI), reducing the need for users to customize the configuration based on the type of job that they are running. A high-level description of pySUMMA is presented in Figure 1. The simulation.py enables the execution of the SUMMA model and, along with file_manager.py, decisions.py, force-file_list, and output_control.py, allows for manipulating SUMMA configuration files. The distributed.py enables the parallel execution of SUMMA.

2.2.4 Data management and transfer

The input data for this study consists of the SUMMA configuration files and the forcing data for the 671 CAMELS basins. The configuration files (e.g., geometries information for the 671 CAMELS basins along with their attributes such as hru_id) are shared within each of the two HydroShare resources holding the Jupyter notebooks. The forcing data are provided in a HydroShare resource (Mizukami and Wood, 2021).

The output files resulting from running the notebooks using the CJW CG and HPC computational environments are: (1) NetCDF output files generated by the SUMMA simulations, (2) a NetCDF file recording the model performance for each basin as measured by the Kling-Gupta Efficiency (KGE) (Gupta et al., 2009), and (3) additional files created by the notebooks such as the figures that visualize the model results.

In the case of the CJW CG environment, after running the notebooks, all files are saved in the CJW CG and are directly accessible to the user. In the case of the HPC environment, the KGE results and other files created by the notebooks (e.g., figures) are automatically transferred to the CJW CG, but the NetCDF output files remain within the HPC environment to avoid transferring large volumes of model output (as a reminder, the size of the model output for the entire VB study was 2.834 TB).

However, if the user of the HPC environment wishes to transfer selected SUMMA NetCDF output files from the HPC to be directly accessible for further analysis and long-term storage, then the CyberGIS-Compute Service (Li et al., 2022) can be used for reliable high-performance large file transfers through the Globus service (Chard et al., 2016; Foster, 2011). As shown in Figure 2, data is transferred from HPC to the CJW using Globus without going through the job submission server. Globus is a software as a service that enables the transfer of datasets of any size between different storage options (personal computers, HPC, etc.) without users being required to be constantly logged in and monitoring the data transfer (Chard et al., 2016). Technically, the CyberGIS-GIS Compute acts as a Globus app client holding a community Globus account that has access to both data endpoints on the Jupyter and target HPC. When data transfer is needed, CyberGIS-Compute initiates a Globus task between the two endpoints and monitors the progress. Users are updated with data transfer status in the notebooks environment during the entire process.

2.3 Model Workflows as Jupyter Notebooks

As mentioned earlier, the model workflows allow the user to reproduce all or subsets of the VB study using either the CJW CG computational resources (referred to later as CJW CG) or the HPC and CJW CG computational resources (referred to later as HPC). The CJW CG and HPC HydroShare resources can be found at Choi et al. (2022b) and Choi et al. (2022c), respectively. The model workflows are documented in three (for CJW CG) or four (for HPC) Jupyter notebooks. Table 1 shows the summary of the steps taken in each notebook, while Figure A2 - A5 show more detailed information for notebooks 1-4. The first three notebooks for both the CJW CG and HPC environments focus on (1) selecting the study basins, simulation period, and model input forcings, (2) running the SUMMA model, and (3) exploring outputs to analyze the effect of each forcing variable in each basin. The HPC computational resource uses a fourth notebook to transfer large unprocessed output data from the HPC to CJW using GLOBUS. Notebooks 1 and 3 are very similar between the two HydroShare resources, and both CJW CG and HPC HydroShare resources use CJW CG computational resources to run these two notebooks. The second notebook differs for the two environments, and the difference is explained in Section 2.3.2. These notebooks assist a modeler in analyzing CAMELS basins individually, providing information on forcings and output variables that are the most/least sensitive in their basin. With some additional work, the CJW CG computational environment can also be hosted on other (non CJW) cloud services, but the HPC environment is more tailored to interact with the CJW cloud service used here.

Table 1. Overview of the notebook 1-4.

#	Notebook Name	Goal	CJW CG or HPC
1	Preprocessing	Prepares forcings, and sets study basins and simulation period	Very similar between HPC and CJW CG environment
2	SUMMA execution	Runs the SUMMA model	Different versions for HPC and CJW CG environment
3	Post-processing	Explores outputs to find out effect of each forcing variable in each basin	Very similar between HPC and CJW CG environment
4	Use Globus to transfer big data	Transfer raw output from HPC to CJW using Globus service	Only for HPC environment

To use the HPC computational resource, the user must obtain access to the HPC by issuing a request through HydroShare to use CJW. Once this access is granted, users are automatically given free access to two alternative HPC resources: (1) the Virtual ROGER (Resourcing Open Geospatial Education and Research) HPC administered by the School of Earth, Society, and Environment at University of Illinois Urbana-Champaign (UIUC) which is integrated with the Keeling compute cluster at UIUC ("Virtual Roger User Guide," 2022) and (2) the Expanse HPC, a much larger NSF XSEDE resource operated and managed by San Diego Supercomputer Center (SDSC) ("Expanse System Architecture," 2022). In theory, the CyberGIS-Compute Service can support other HPCs as well, but we did not test other HPCs. In this study, among the provided HPC options, we only used Expanse to demonstrate the cyberinfrastructure: in our initial experiments Expanse HPC performed faster than Virtual ROGER and the goal here was to show how a HPC can scale up a study by speeding up the modeling process compared to a non-HPC environment rather than an inter-comparison between different HPCs. Users who do not wish to use HPC computational resources can use CJW CG computational resources directly to run smaller modeling jobs.

The hardware specifications of the CJW CG and the Expanse HPC are compared in Table 2. The CJW CG has only 3 compute nodes each of which has eight CPUs with 1.996 GHz Clock Speed and 30 GB DRAM. Each user can only use up to six CPUs and the CPUs can be shared among users. This means the maximum degree of parallelism for simulations using this computational resource is six. Thus, in case of running one basin from the VB study (704 runs) and using all the six available CPUs, each CPU will need to run 117.33 simulations (some of them 117 and others 118 simulations). The Expanse HPC has 728 AMD Rome standard compute nodes each of which is equipped with 256 GB DRAM and 128 2.25 GHz CPUs ("Expanse User Guide," 2022). The Expanse HPC allows the user to only use up to 2 nodes at a time, i.e., 256 CPUs or the maximum degree of parallelism for simulations. Thus, if a user is running one basin from the VB study (704 runs) and using all the available 256 CPUs, then each CPU will need to run 2.75 simulations (some of them 2 and others 3). This shows how the HPC resource can scale up the model runs offering a high-performance tool. More details about the run-time performance of the notebooks are discussed in the results and discussion section.

Table 2. Hardware specifications of the computational environments.

Computational Environment	Node count	Number of CPU cores per node (for parallel runs only)	Clock Speed (GHz)	DRAM/node (GB)
CJW CG*	3	8	1.996	30
Expanse HPC**	728	128	2.25	256

*AMD EPYC-Milan Processor. Each user can only up to 6 CPUs and the CPUs can be shared among users.
**AMD Rome Standard Compute Nodes. Each user can only use up to 2 nodes, which means 256 CPUs, the maximum number of parallelism for simulations.

The following subsections discuss the general purpose of each notebook used to reproduce parts of the VB study. For specific coding details, refer to the notebooks in the HydroShare resources at Choi et al. (2022b) and Choi et al. (2022c).

2.3.1 Data processing notebook

The first notebook (JN 1: Preprocessing) processes the original CAMELS SUMMA files and the input forcing datasets (Table A2). The user can select one or more CAMELS basins (1-671 basins) but by selecting a higher number of basins the computational time and expense increases. Notebook 1 subsets the original CAMELS SUMMA files, producing SUMMA attributes, parameters, initial conditions, and hourly NLDAS forcing files for the selected basin(s). Then, additional forcing datasets for the hydrologic model sensitivity study are developed from the NLDAS data files (FORCINGS box in Figure A1) as discussed below.

For each SUMMA-model setup, variations in 14 SUMMA-generated outputs, described in Table A1, are examined with respect to variations in seven input forcings (air pressure (*prs*), air temperature (*tmp*), long wave radiation (*lwr*), precipitation rate (*ppt*), specific humidity (*hum*), shortwave radiation (*swr*), and wind speed (*wnd*)), under different model parameterizations and configurations. The SUMMA outputs generated with the one-hour NLDAS forcing dataset are considered the benchmark (NLDAS dataset 1; FORCINGS box in Figure A1). The rest of datasets (*ppt* to *prs* datasets; FORCINGS box in Figure A1) are developed, holding each of the individual forcing variables constant over a 24-hour period while the other six forcing variables contain the original hourly NLDAS values.

Figure A2 shows the steps taken in the first notebook. This notebook is the same for the CJW CG and HPC environments except that the simulation time period and basins to be explored are pre-populated differently. The user can change these setups in the third step of this notebook (step 1_3). In the last step of this notebook, users can visualize the individual forcing variables held constant over a 24-hour period against the original hourly NLDAS values using hourly and cumulative plots.

2.3.2 SUMMA execution notebook

The second notebook (JN 2: Running SUMMA) executes the SUMMA model using the input data from the first notebook for four different sets of SUMMA basin runs, outlined in Figure A1 (RUNS box) and described in detail in The VB study. The first set of basin runs (DEFAULT; 8 SUMMA runs per basin; RUNS box) uses the eight forcing datasets (FORCINGS box) combined with default parameters and a default SUMMA configuration. The SUMMA default configuration is set in the resource model decision file.

The second set of basin runs (LHS; 88 SUMMA runs per basin; RUNS box in Figure A1) uses the eight forcing datasets combined with 11 parameter sets and a default SUMMA configuration. The 11 parameter sets consist of the default parameter set and 10 additional parameter sets with 15 commonly calibrated parameters (Table A2). As detailed in the VB study, the parameters are sampled using Latin Hypercube Sampling (LHS) over their defined range. The *pyDOE LHS* function (Lee, 2014) is used to create unique 10 x 15 LHS sampling matrices for the selected basin. Then the LHS matrices are used to produce 10 parameter sets of the 15 parameters while considering the parameter constraints listed in Table 2. The choice of a different seed value will lead to different LHS sets (and these sets will be different from the ones used by the VB Study).

The third set of basin runs (CONFIG; 64 SUMMA runs per basin; RUNS box in Figure A1) uses the eight forcing datasets combined with the default parameter set and eight SUMMA configurations. The eight SUMMA configurations, outlined in the CONFIGURATIONS box in Figure A1, test three model decisions (stomatal resistance (stomResist), choice of snow interception parameterization (snowIncept), and choice of canopy wind profile (windPrfile) with two options for each decision. Note the default configuration for this study is shown in bold in the CONFIGURATIONS box in Figure A1:BallBerry, lightSnow, and logBelowCanopy.

The fourth set of basin runs (COMPREHENSIVE; 704 SUMMA runs per basin; RUNS box in Figure A1) includes the DEFAULT, LHS, and CONFIG basin runs, and is the only set that needs to be run to replicate a single basin sensitivity study following the VB study method (six years of simulation must be run for replication). For testing purposes, sets 1-3 can also be run by themselves. The 10 parameter set files for the basin from the LHS sampling plus the default parameters (11 parameter sets) are run each with eight SUMMA configurations (CONFIGURATIONS box in Figure A1).

Figure A3 shows the steps taken in the second notebook. The first two steps in this notebook are the same for the CJW CG and HPC environments but the rest of the workflow differs. In the CJW CG notebook, the user can define the simulations by selecting the simulation period, model configuration, and/or parameter values. Depending on which run complexity choice (i.e., DEFAULT, LHS, CONFIG, COMPREHENSIVE in the RUNS box in Figure A1) is selected the notebook executes a specific set of code cells using a conditional statement logic (e.g., if user selects *config_prob == 1*, step 2_7 is run which leads to CONFIG runs as shown in the RUNS box in Figure A1). Users need to carefully consider the number of basins and the length of the simulation period as the CJW CG environment is not powerful enough to run large simulations in a reasonable time. In the HPC notebook, we only provided the user with the option to run the most complex problem, i.e., *lhs_config_prob*, as the HPC is powerful enough to run the full problem making it unnecessary to allow for simpler problems. The user can still change the simulation period (in step 2_3 of the workflow in Figure A3). The other main difference between the CJW CG and HPC notebooks is that the codes calculating KGE values for the HPC notebook are executed on the HPC (Step 2_8 in HPC branch in Figure A3) while for the CJW CG environment, the KGE values are calculated locally on CJW CG (Step 2_9 in CJW CG branch in Figure A3). In the HPC environment, the KGE values are calculated on the HPC resource to prevent having to transfer large data volumes from the HPC to the CJW CG with the sole purpose of calculating performance metrics. Users can use Globus to transfer selected output files from HPC to the CJW CG for additional analysis. Notebook 4, which exists only in the HPC environment, was developed for this purpose and is discussed in section 2.3.4.

A modified and scaled (range between -1 and 1) version of the KGE was used as an indicator of model output sensitivity to a change in input forcing based on the work of Clark et al. (2021) and Mathevet et al. (2006) and is described in the VB study. The KGE test compares hourly model outputs generated with the benchmark forcing dataset (NLDAS dataset 1; Table A2) with outputs generated with the forcing datasets with one forcing held constant (CNST datasets 2-8; Table A2). KGE values are ranked from low to high to determine relative order of forcing influence on model outputs with highest rankings associated with least influence of change to 24-hour constant forcing.

2.3.3 Post-processing notebook

The third notebook (JN 3: Post-processing) produces visualizations of the sensitivity of SUMMA model output to the temporal resolution of the model forcing. Figure A4 shows the steps taken in the third notebook. The notebooks for CJW CG and HPC environments are the same. For the selected basin(s), eight plots are generated with Notebook 3 that follow the analysis in the VB study. The reader is referred to the supplementary materials and the VB study for a detailed explanation of each of the eight plots. In this paper, we only present the second figure generated by Notebook 3, i.e., KGE values for each output variable for all 8 DEFAULT model runs.

2.3.4 Model output transfer

The fourth notebook (JN 4: Use Globus) is only included in the HPC resource (Figure A5) to transfer SUMMA output files from HPC to CJW on HydroShare. To retrieve the data from the HPC, this notebook needs a job ID submitted to the HPC and created in Notebook 2. While this notebook is running users can see the live status of the file transfer managed by the CyberGIS-Compute Service. Once running of this notebook is successfully finished, the user will be able to see the location of the transferred file on CJW.

2.4 Performance analysis

We tested the performance of the cyberinfrastructure using a number of model scenarios, using six years of simulation (to be consistent with the VB study) and varying the number of studied basins for each computational environment, described in Table 3. For the CJW CG environment, we tested the performance of notebooks 1-3 for three scenarios (Table 3, rows 1 - 3): (1) one basin (a total of six years of simulations), (2) four basins (a total of 24 years of simulations), and (3) six basins (a total of 36 years of simulations). We decided not to test the CJW CG environment for more basins as the CJW CG runs were slow and the HPC resource was available for larger simulations.

For the HPC environment, we used Expense HPC, and tested the performance of notebooks 1-3 for 12 scenarios (Table 3, rows 4 - 15). In these scenarios, we varied the number of allocated CPUs (128 or 256) for parallelism and the total number of basins ranging from one basin (a total of six years of simulations) to 20 basins (a total of 120 years of simulations, which equals about three percent of the total simulation years for the whole VB study). To test the performance of Notebook 4, transferring output files from HPC to the CJW, we only used scenarios HPC_256_1 to HPC_ 256_6 (rows 4 - 9 in Table 3) and repeated each transfer 5 times to obtain a range of run-time for each of the scenarios.

Table 3. Model scenarios for notebooks run-time performance analysis.

Row	Model scenario name	Number of CPU cores allocated	Number of basins	Simulation years	Total simulation years
1	CJWVM_1	6	1	6	6
2	CJWVM_2	6	4	6	24
3	CJWVM_3	6	6	6	36
4	HPC_256_1	256	1	6	6
5	HPC_256_2	256	4	6	24

6	HPC_256_3	256	6	6	36
7	HPC_256_4	256	10	6	60
8	HPC_256_5	256	15	6	90
9	HPC_256_6	256	20	6	120
10	HPC_128_1	128	1	6	6
11	HPC_128_2	128	4	6	24
12	HPC_128_3	128	6	6	36
13	HPC_128_4	128	10	6	60
14	HPC_128_5	128	15	6	90
15	HPC_128_6	128	20	6	120

3 Results and Discussion

In this section, we first briefly present results of the modeling case study that served as a motivating use case for the cyberinfrastructure. Then, we present results of the performance analysis focusing on contrasting the CJW CG and HPC notebooks using a variety of model setups. Then, we summarize the resulting resources from this study that are shared on HydroShare. Finally, we discuss the resulting system including opportunities and challenges identified through this research that can be the focus of future research.

3.1 Results of the modeling case study

Four CAMELS basins with diverse characteristics (Table 4) were chosen as examples of the effect of basin characteristics on model results. We specifically selected these four basins for this modeling case study because we found that they all show different patterns. For the four selected basins, Figure 3 shows the KGE values for each SUMMA output variable using the DEFAULT (BIL; CONFIGURATIONS box in Figure A1) model configuration runs. The runs consist of one reference simulation in which all forcing variables vary on an hourly basis (NLDAS dataset 1; FORCINGS box in Figure A1) and seven simulations in which one forcing variable is held constant at the mean daily value throughout each day (the seven datasets *ppt* to *prs*; FORCINGS box in Figure A1). KGE values were calculated relative to the reference simulation for each of the seven simulations using five years of hourly model output from 10/1/1991 - 9/30/1996.

Table 4. Basin descriptions for individual basin analysis.

USGS Station ID	**Name**	**CAMELS Attributes**						
		Drainage area (km^2)	Gage datum (m)	Mean daily precipitation (mm/day)	Fraction of precipitation falling as	Aridity	Mean daily discharge (mm/day)	Runoff ratio*

					snow			
01632900	Smith Creek Near New Market, VA	242	268	2.91	0.10	0.89	0.80	0.27
02212600	Falling Creek near Juliette, GA	187	1202	3.37	0.01	1.19	0.74	0.22
09378630	Recapture Creek Near Blanding, UT	10	2195	1.58	0.50	0.50	0.21	0.13
11264500	Merced River at Happy Isles Bridge near Yosemite, CA	469	1228	2.64	0.91	1.15	1.94	0.73

* Annual runoff / annual precipitation

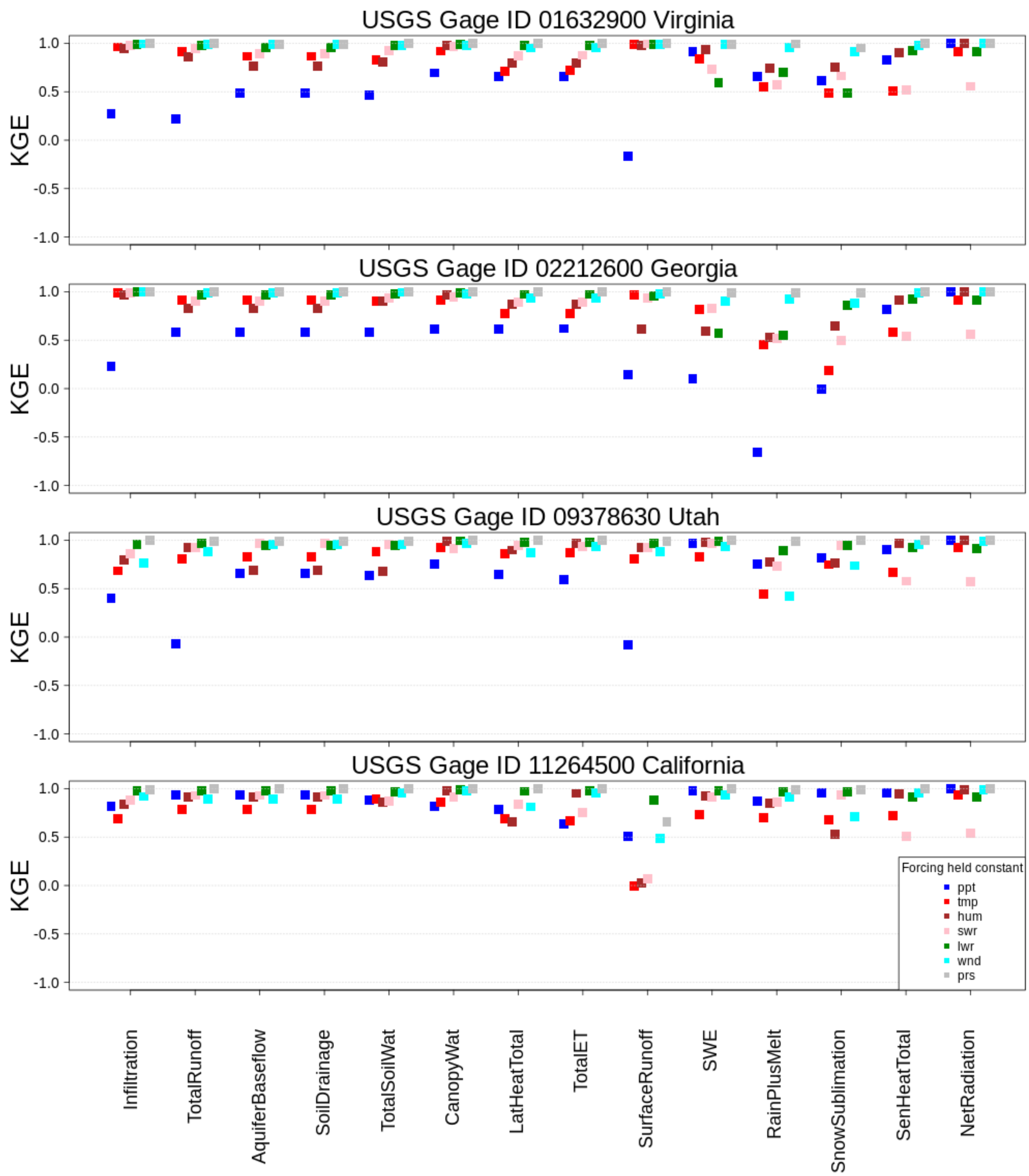

Figure 3. KGE values using the DEFAULT model runs for each CNST dataset (datasets 2-8; Table A2), grouped by SUMMA output variable.

Figure 3 demonstrates the variability in model output sensitivity to the temporal resolution of the forcing variables. The first three basins (gages 01632900, 02212600, and 09378630) show a strong *ppt* temporal aggregation influence using DEFAULT, whereas gage 11264500 is more influenced by *tmp*, *hum*, and *swr* temporal aggregation. In other words, a higher temporal resolution is necessary for the aforementioned forcing variables in the given basins to capture the sub-daily hydrologic response shown by the reference simulation. The weaker influence of *ppt* temporal aggregation on the gage 11264500 compared to other gages can be attributed to its high fraction of precipitation falling as snow, 0.91 as opposed to 0.1, 0.01, 0.5 (Table 4).

Also in Figure 3, we see varying ranges in KGE values for particular output variables. As an example, SurfaceRunoff is affected by constant hourly values of *ppt* for gages 01632900 and 09378630; *ppt* and *hum* for gage 02212600; and *tmp*, *hum*, *swr*, *wnd*, *ppt*, and *prs* (most to least dominant) for gage 11264500. This shows the forcing variables in each basin that need to have a higher temporal resolution to reproduce the SurfaceRunoff output in the reference simulation. In this section, we only presented one example of an inter-basin comparison to illustrate how different the results can be across different basins. Researchers can further explore the differences between individual basins using other plots that can be made using the interactive Jupyter notebooks, and also reproduce the results from the original VB study.

3.2 Results from performance analysis

Figure 4 shows the run-time for the data processing notebook (Notebook 1) and the post-processing notebook (Notebook 3) for the 15 scenarios listed in Table 3. Notebooks 1 and 3 are very similar between CJW CG and HPC computational environments. Notebooks 1 and 3 do not take a significant time to run because they are only preprocessing and output analysis notebooks, and no simulations are run. For scenarios with fewer than 30 simulation years, Notebook 1 takes longer than Notebook 3, but this changes for scenarios with more simulation years as the rate of run-time increase with simulation years is much higher with Notebook 3 than with Notebook 1. For the CJW CG environment, the average time to run Notebooks 1 and 3 across the tested scenarios only takes 0.6% of the entire time needed to run all Notebooks 1, 2, and 3. This means the time required to run data processing and post-processing notebooks is not a limiting factor for running the simulations. For the HPC environment, this ratio increases to 8.5% and 11.3% when using 128 and 256 CPUs, respectively. This dramatic increase in the ratio is due to the significant decrease in run-time of Notebook 2 when using HPC.

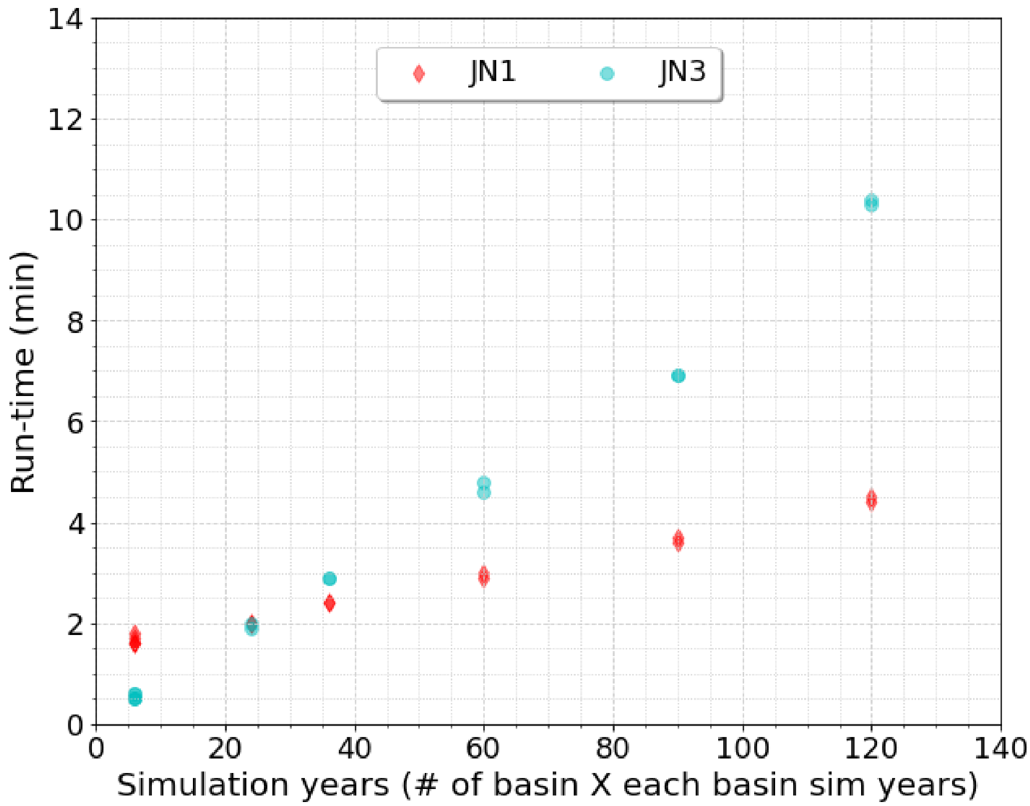

Figure 4. Notebook 1 (JN1) and 3 (JN3) run-time performance analysis for different model simulations (both JN1 and JN3 were run on CJW CG no matter whether the HPC or CJW CG environment was used for the modeling; therefore, we do not distinguish between the environments in this figure).

The run-time for the SUMMA execution notebook (Notebook 2) for the 15 model scenarios using different computation environments is shown in Figure 5. The high rate of run-time increase with increasing simulation years for the CJW CG environment emphasizes that while the CJW CG environment is technically able to simulate smaller models, it might not be fast enough to run larger simulations. In the case of running six basins for six years, the HPC was 3.6 and 2.6 times faster than the CJW CG, when using 256 and 128 CPUs, respectively. HPC with 256 CPUs (scenario HPC_256_6) could finish the simulations for 120 years (3% percent of the VB study) in 2 hr and 10 min while HPC with 128 CPUs (scenario HPC_128_6) could run the same problem in 1.48 times of the time need by HPC_256_6. Using the HPC with 256 CPUs, assuming a conservative linear extrapolation, the SUMMA simulations from Notebook 2 are expected to be done in about 75 hours for the entire VB study. In summary, HPC provides considerably faster simulations making them ideal to run for larger studies.

When using the HPC resource and in the case of 120 years of simulation, dividing the number of the allocated CPUs by two led to about a 50% increase in the run-time and not 100% as one might expect. This non-linear scaling can be mainly attributed to 1) communication overhead in the computational resource that reduces scaling, and 2) the fact that some parts of the codes in

Notebook 2 did not utilize parallelism. For example, KGE values were only calculated after they were exported as NetCDF files instead of being calculated directly from the raw SUMMA output files. The rate of run-time increase for HPC with 128 CPUs is higher compared to that for HPC with 256 CPUs. This may be attributed to the communication overhead because each CPU in the case of the HPC with 128 CPUs needs to run twice as many simulations compared to HPC with 256 CPUs.

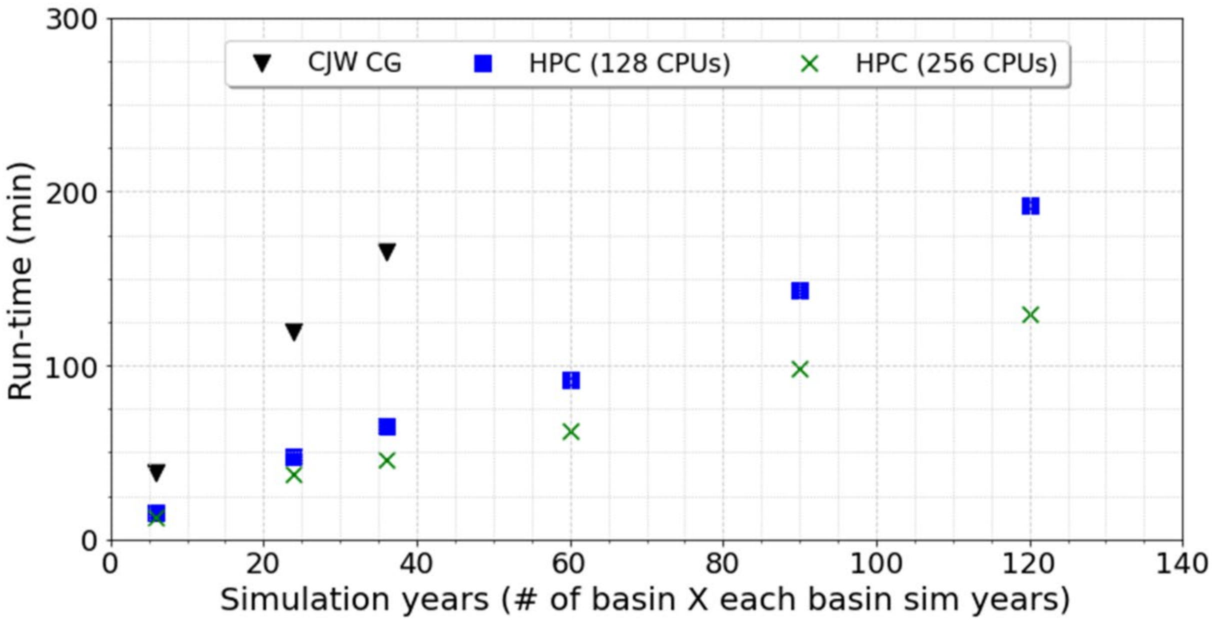

Figure 5. Notebook 2 run-time performance analysis for different model simulations using the CJW CG, or HPC (Expanse with 256 or 128 CPUs) options.

The run-time for transferring the SUMMA output files from Expanse HPC to CJW on HydroShare using the Globus service integrated by CyberGIS-Compute Service is shown in Figure 6. Each transfer was repeated 5 times to obtain a range of run-time for each of the model simulations with a different total number of simulation years. The range of the transfer time for each total number of simulation years is small, indicating a consistent data transfer. For 120 years of simulation, it took 14.5 min on average to transfer 118 GB of data from HPC to CJW, highlighting that the data transfer approach from HPC to CJW is fast and stable. The transfer rate (GB/min) is independent of data size (Figure 6).

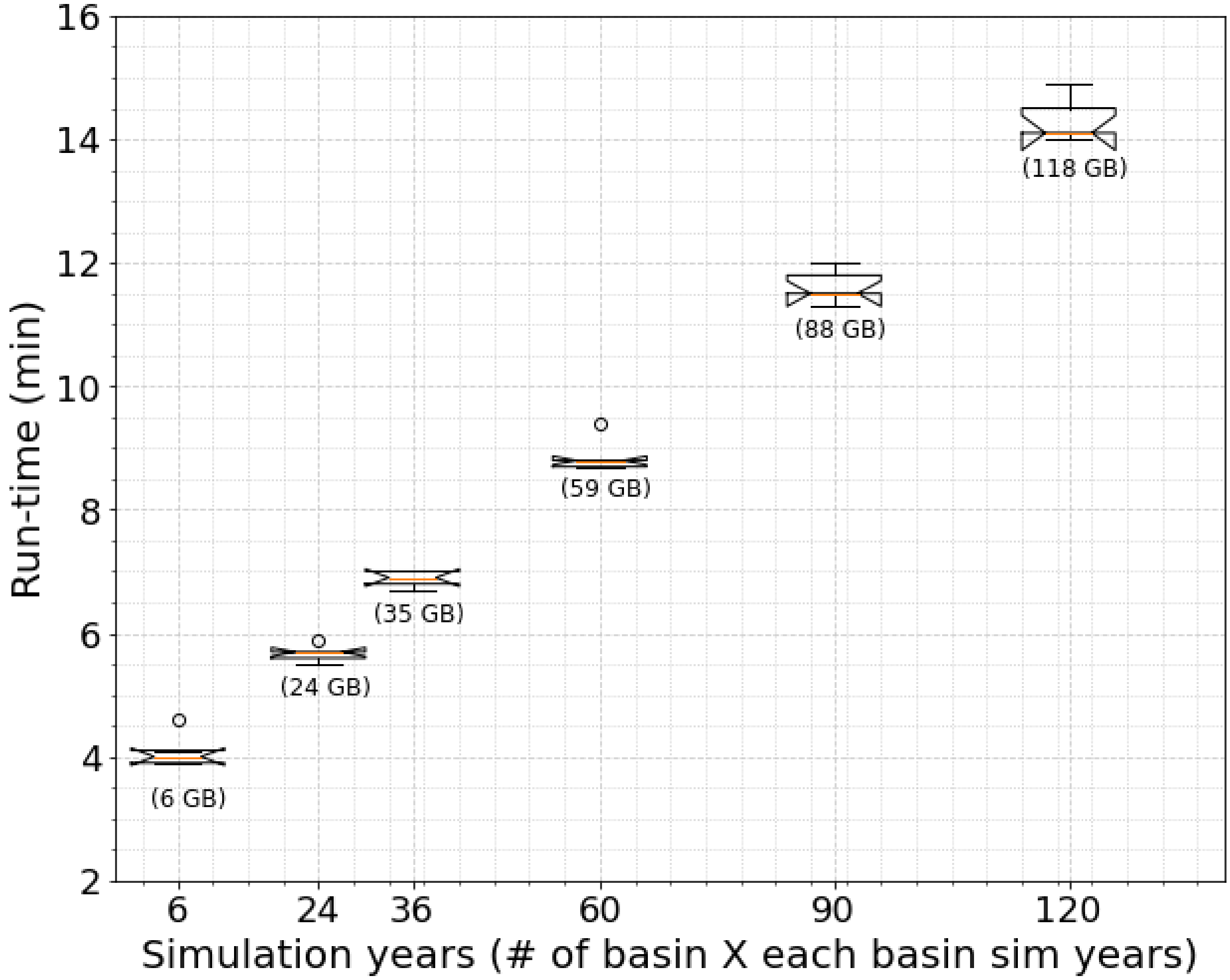

Figure 6. Boxplots for Notebook 4 run-time performance analysis for five different simulation years to transfer data from Expanse HPC to CJW on HydroShare. Each transfer was repeated five times to obtain a range of run-time for each of the model simulations with a different total number of simulation years.

3.3 Data organization in HydroShare

The data for this study was pre-processed and the output post-processed by using existing Python packages. The study demonstrates the potential for using the online repository of HydroShare to not only store data and modeling code, but to also store computational environments, API version documentation, and container installation. HydroShare, as a hydrology-based repository service, facilitated this by allowing all the parts of the problem to be stored together as one resource. Furthermore, parts of the resource can be extracted and made into a new version of the resource (updated, revised, or modified), to promote collaboration.

To this point, a HydroShare collection resource was created that contained three composite resources. These resources are published and have Digital Object Identifier (DOI) which makes them immutable and findable. Figure 7 shows the landing page for the HydroShare collection resource that groups the three composite resources. The three composite resources that are contained by this collection resource are shown in dialogue box 1, the "Related Resources" in box 2 refers to this paper, and box 3 shows the information on how to cite this resource. Figure 8 shows the landing page for the HydroShare composite resource holding the HPC notebooks. Box 1 shows the contents of the resource, most importantly the four Jupyter notebooks and the readme.md file. The readme.md file (box 2) provides the user with the instructions on how to run the notebooks. Box 3 shows the information on how to cite this HydroShare resource.

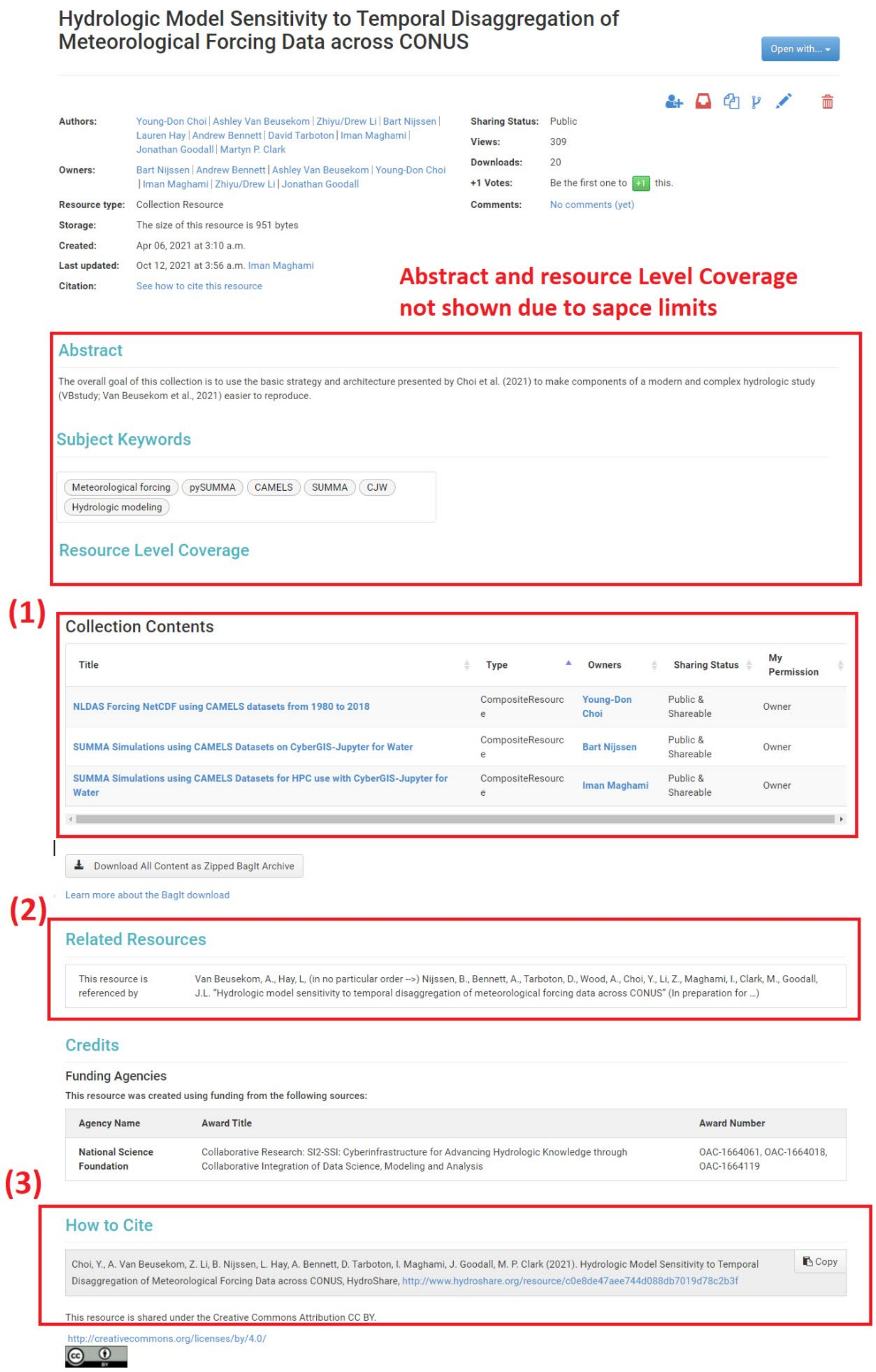

Figure 7 The HydroShare landing page for the collection resource developed by this study (Choi et al., 2022a).

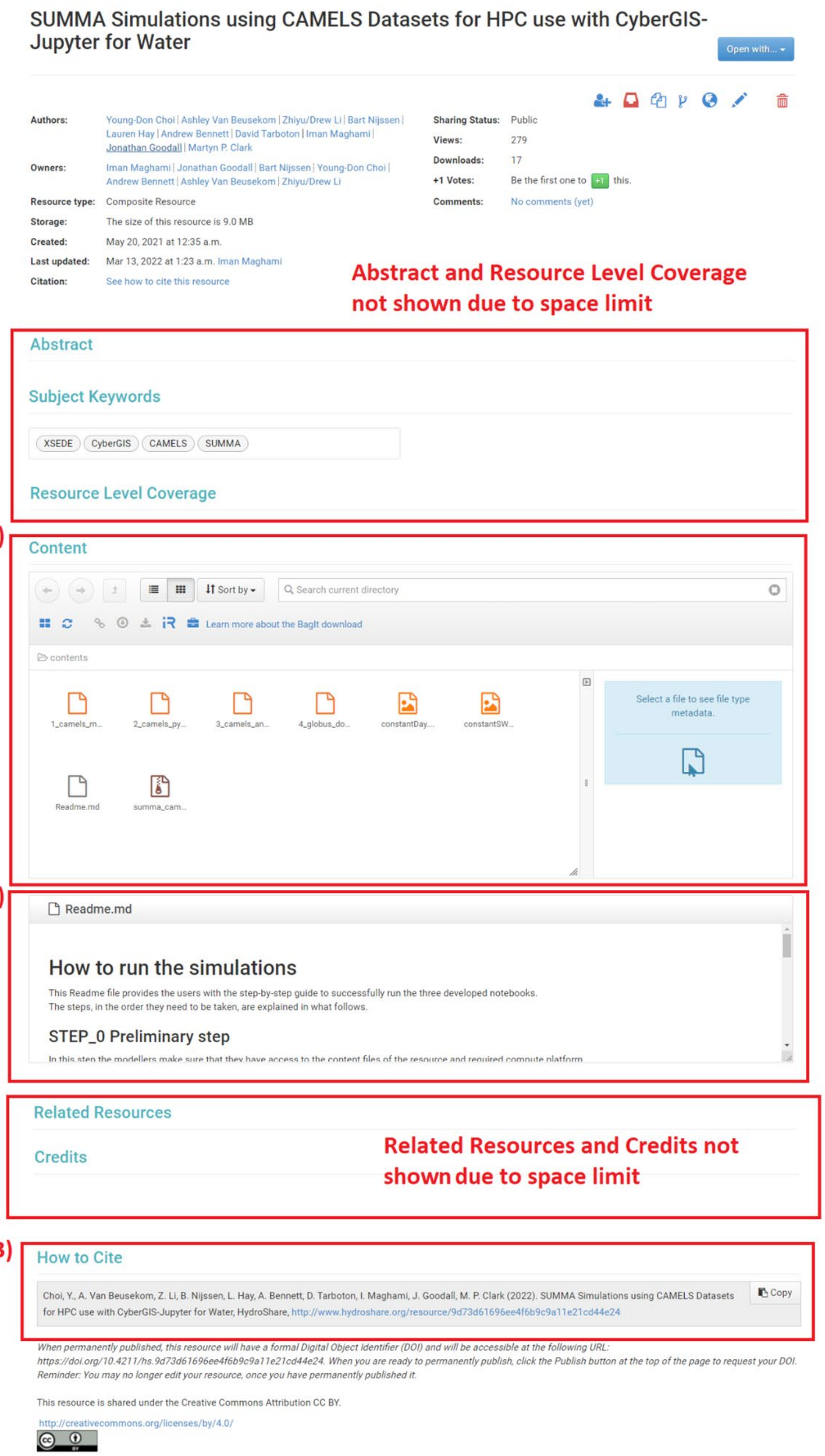

Figure 8 The HydroShare landing page for the HPC resource developed by this study (Choi et al., 2022c).

3.4 Opportunities and challenges

This study demonstrated a real-world working implementation application of strategies for reproducible hydrologic modeling presented by Choi et al. (2021) to a large-scale hydrologic study (the VB study). This section discusses the opportunities and challenges of this implementation. If one needs to adopt this cyberinfrastructure for studies significantly differing from the VB study, considerable changes or extra steps might be needed. For instance (1) if exploring non-CAMELS basins, then extra steps to prepare the inputs might be needed, or (2) if using hydrologic models other than SUMMA, then containerization of the model might be needed. Despite the plausible challenges when making these non-trivial extra steps, the intended main opportunity here is that the modeling community can learn from the presented open cyberinfrastructure considering the commonalities among the hydrologic models with regard to the input data, preprocessing, processing, and postprocessing steps needed by them (Knoben et al., 2022).

Minimal changes in the notebooks are required to use the presented cyberinfrastructure to rerun parts or all of the VB study or to extend the experiments performed in that study for selected CAMELS basins. With these minimal changes, a user could use (1) different CAMELS basins, (2) different parameters in the LHS set, (3) different simulation periods, e.g., a drought period, (4) more than 10 LHS sets, e.g., a more thorough exploration of the parameter space, and (5) additional SUMMA model configurations. The last two changes, i.e., using a larger number of LHS sets and different model configuration/decisions, highlights a major challenge in reproducing a computationally complex study. Here, the limit on manageable data size was pushed, even when running a few basins. HPC computational power was required to run the full six years of simulation; expanding the parameter exploration space or adding model decisions would compound the data size. Thus, while this work is advancing cyberinfrastructure used for big data in hydrology, challenges remain.

The second major challenge that is encountered is implementing version control. What if users need to run the Jupyter notebooks presented in this study in their own computational environment (not deployed on CJW), or they need to install a newer version of a model API? How can they make sure they have a reproducible framework that is robust enough to tackle the version control problem? Because there are many individual pieces of software, it was challenging at times for the study team to keep all the software versions synchronized. We propose that future research should tackle the version control challenge by making the computational environment all documented and installable via a Python environment file. The pySUMMA code, which is used for hydrology modeling, was installed via conda just as the rest of the infrastructure. In the future, Python package updates will break compatibility, but compatibility can be preserved by installing the older versions (as documented in the environment file), or the user understanding the updates in order to manually work around the updated package incompatibility. If a researcher wants to use a newer (future) version of pySUMMA, then they may need to debug some parts of the Jupyter notebooks that are affected by the changes. While this is not an ideal way to handle version updates, at least the researcher has options of a working, albeit older, computational environment, from which to begin reproducing the study before updating to newer software.

The specifics of the environment can be placed in a Python environment.yml file that can be shared as part of the online model and data repositories, and can be installed with an installation notebook inside the repository. This can use best practice for transparency about what dependencies the computational gateway interface notebooks need to run. The specifics of each dependency can be described in the installation notebook, so that if in the future there are issues with the availability of that dependency, then a suitable substitute can be found. Version control issues can be thus addressed through this methodology, albeit an imperfect solution depending on possible user troubleshooting.

In addition to the two major challenges described above, there are two additional challenges related to the use of the HPC environments: 1) large data transfers between computational environments, online data repositories, and a user's personal computer and 2) allowing users to execute their workflows on different HPC environments based on their use case and access to HPC environments. There may be cases, for example, where users does not want to utilize HPC resources due to financial cost concerns and need to transfer a large amount of model outputs from an HPC environment's temporary scratch directory to a Jupyter compute environment to further analyze the data using the Jupyter compute environment. Transferring large datasets, e.g., the entire output from VB study or even the four selected basins study explored in this paper, would be slow and unreliable using standard data transfer approaches, i.e., compress data into a big package and then transfer it. In this study, we used Globus to do this data transfer which can transfer multiple individual files in parallel without a need to compress data a big package, and other related cyberinfrastructures that do not currently use Globus or a related technology could benefit from doing so. Globus is not limited to data transfers between the HPC environment and the Jupyter compute environments (CJW in the case of this study), however. In fact, it is possible that the full or a large portion of the model output can be stored on an online data repository or even on a user's own personal computer. In either case, the online data repository or the user's personal computer, the outputs could be downloaded using Globus if Globus is installed, and they become a Globus server. Making a user's personal computer a Globus server may be the case that the user prefers to back up a model run not in an online data repository but at some other location. In this case, Globus could be used to connect directly with the HPC environment thereby bypassing both any Jupyter compute environments (CJW in the case of this study) as well as online data repositories (HydroShare in the case of this study) as an intermediate storage location. If the large data takes much of the space in the user's personal computer, user may consider transferring it to external hard drives that offer larger capacity. To allow users to execute their workflows on different HPC environments, users would need to set up their own job submission service and configure the Jupyter environment (e.g., CJW) to the specific HPC environment that they have access to. Although the job submission software used in this study is open source, it is customized for the UIUC HPC used in the study, so it cannot be directly used for other HPCs. Future work could be for CJW to act as a connector to user supplied HPC environments. In this case, CJW would ask users to provide their own credentials and to their own HPC, rather than only using the UIUC HPC service. While not a simple task, standardization of job submission approaches across HPC environments makes this functionality possible. Generalizing the approach through future research could benefit users to access their own institutional HPCs and other HPCs at the national level that the user has access to.

4 Conclusions

The importance of reproducibility is broadly recognized across different scientific disciplines. When it comes to computational hydrology, this can be a significant challenge. This research shows how an architecture that integrates the (1) online data repositories, (2) computational environments, and (3) model API can facilitate reproduction of the components of modern and complex hydrologic studies. For this purpose, we used a recently published large-scale hydrologic study (VB study) as an example. We designed and built cyberinfrastructure that utilized software components to enable intuitive, and online access to computational environments. This approach was used to remove the potential software inconsistencies from users' differing personal software editions, as well as to make implementation easier with pre-compiled software, with the added complication of a computationally expensive research problem instead of a case study. This approach gave the user the option to use either the CJW CG or HPC computational environments, depending on how much they need to reproduce a problem more representative of the big-data problem. Using HydroShare as the data repository, and containerization of the pySUMMA API (with Docker or Singularity in the case of the HPC environment) along with a computational gateway interface of Jupyter notebooks both hosted on the CJW made this possible. Three Jupyter notebooks for the CJW CG environment and four Jupyter notebooks for HPC environment were developed. Notebooks 1-3 for both CJW CG and HPC environments enable, (1) preparing the forcing data, simulation period, and study CAMELS basins, (2) executing SUMMA hydrologic model, and (3) visualization of the results. Notebook 4, only developed for the HPC environment, enables transferring large data from HPC to the scientific cloud service (i.e., CJW) using Globus service integrated by CyberGIS-Compute in a reliable, high-performance and fast way.

We presented a modeling case study subset from the VB study that served as a motivating use case for the cyberinfrastructure. The case study showed how four individual basins with different characteristics can lead to different patterns of temporal aggregation for each of the forcing variables given the same model setup. The case study served to show that the developed cyberinfrastructure enables others to reproduce the VB study for subsets of the original domain as a basis for doing additional research enabling conclusion-reproducibility beyond bit-reproducibility.

We analyzed performance of the notebooks focusing on contrasting HPC and CJW CG notebooks using a variety of model scenarios. The HPC environments could perform significantly faster simulations compared to CJW CG, enabling users to explore a large number of basins and simulation periods. This clearly showed how the use of HPC from a Jupyter gateway could advance the reproducibility of modern and complex hydrologic studies. The run-time performance analysis for the big data transfer notebook for the HPC environment showed that the method used was stable, reliable and fast. Therefore, similar studies could easily benefit from the same approach for transferring large data between scientific cloud services.

With the focus of this research was on conclusion-reproducibility over bit-reproducibility of the VB study, users can easily modify the notebooks to test different situations by varying the study basins and periods, parameterizations, and model configurations. These situations highlighted two major challenges. First, the complexity of the big-data problem eventually became large enough that it needed to be run using the HPC computation environment, which presented other smaller challenges of data transfer and portability of the HPC environment. Second, implementation of a version control system was needed (e.g., when a user needs to install a newer version of a model API or when a user needs to run these codes on their local machine rather than the used cloud-based computational environment). Sharing the dependencies of the computational environments as a Python environment yml file and an installation notebook that installs them was discussed as a future solution to tackle the version control issue.

Finally, as a broader impact, the VB study methodology replicated with interactive codes could also serve as a valuable educational resource, allowing educators to present sophisticated modeling experiments for use within classrooms through online Python notebooks. Likewise, the basic approach could be extended to enable new water decision-support systems that take advantage of the SUMMA framework and HPC yet remain easy to interact with through notebooks. This can help to, for example, evaluate forcing sensitivity to a water resources management objective, or explore the parameter and model uncertainties of SUMMA using different algorithms such as Markov chain Monte Carlo (MCMC), and Bayesian model averaging (BMA) (Samadi et al., 2020) in a systematic manner. With more work to harden and improve the usability of the system presented here, these additional use cases can be possible.

Declaration of Competing Interest

The authors declare no conflicts of interest relevant to this study.

Acknowledgments

This work was supported by the National Science Foundation (NSF) under collaborative grants 1664061, 1664119 and 1664018 for the development of HydroShare (http://www.hydroshare.org), 1928369 for the integration of Reproducibility methods into HydroShare. The work also was supported by the Institute for Geospatial Understanding through an Integrative Discovery Environment (I-GUIDE) that is funded by NSF under award No. 2118329. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

Appendix.

This section provides supplemental material to support our methods and results. The figures and tables are referred to in the main text.

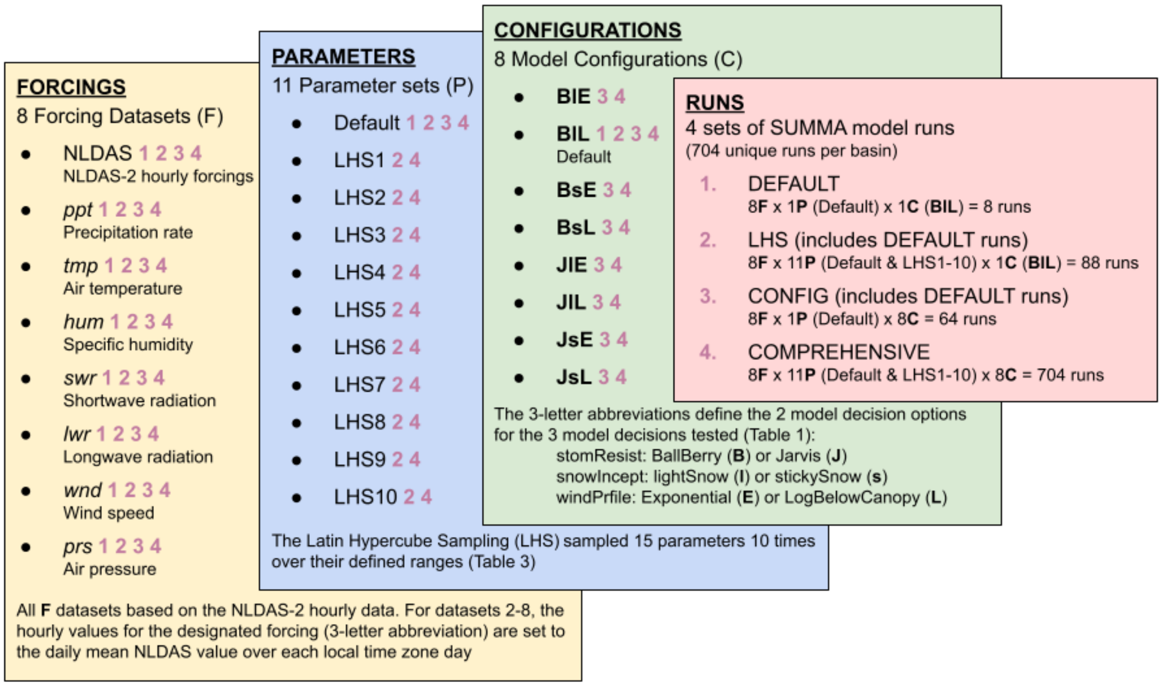

Figure A1. An overview of the forcing datasets (FORCINGS; yellow box), parameter sets (PARAMETERS; blue box), and model configurations (CONFIGURATIONS; green box) used in the 704 SUMMA model runs (RUNS; pink box) performed for each of the 671 CAMELS basins. Note the pink numbers that follow each forcing, parameter, and configuration refers to the SUMMA model run set as numbered in the pink RUNS box (e.g., the Default parameter set in the PARAMETERS box is used with SUMMA model runs 1, 2, 3 and 4 in the RUNS box) (source: modified from Van Beusekom et al., 2022).

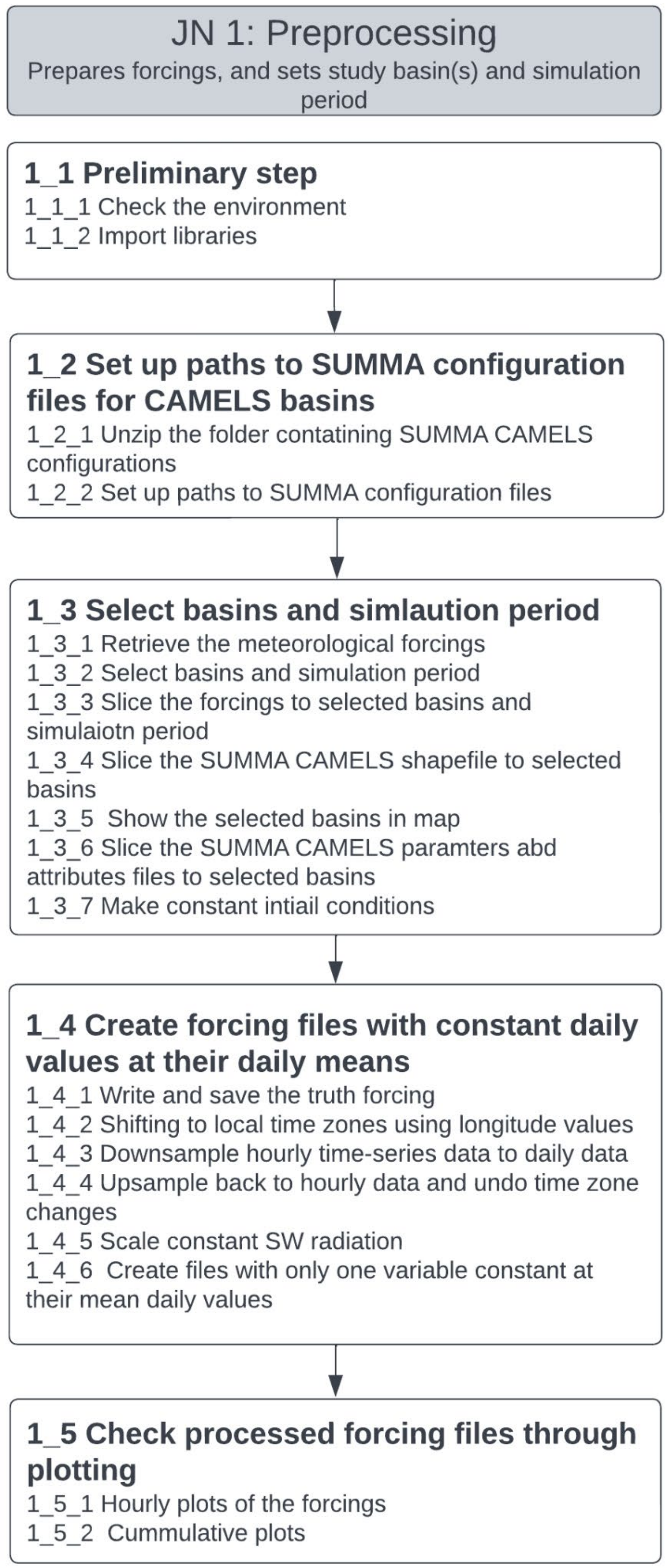

Figure A2. The preprocessing notebook (JN1) diagram.

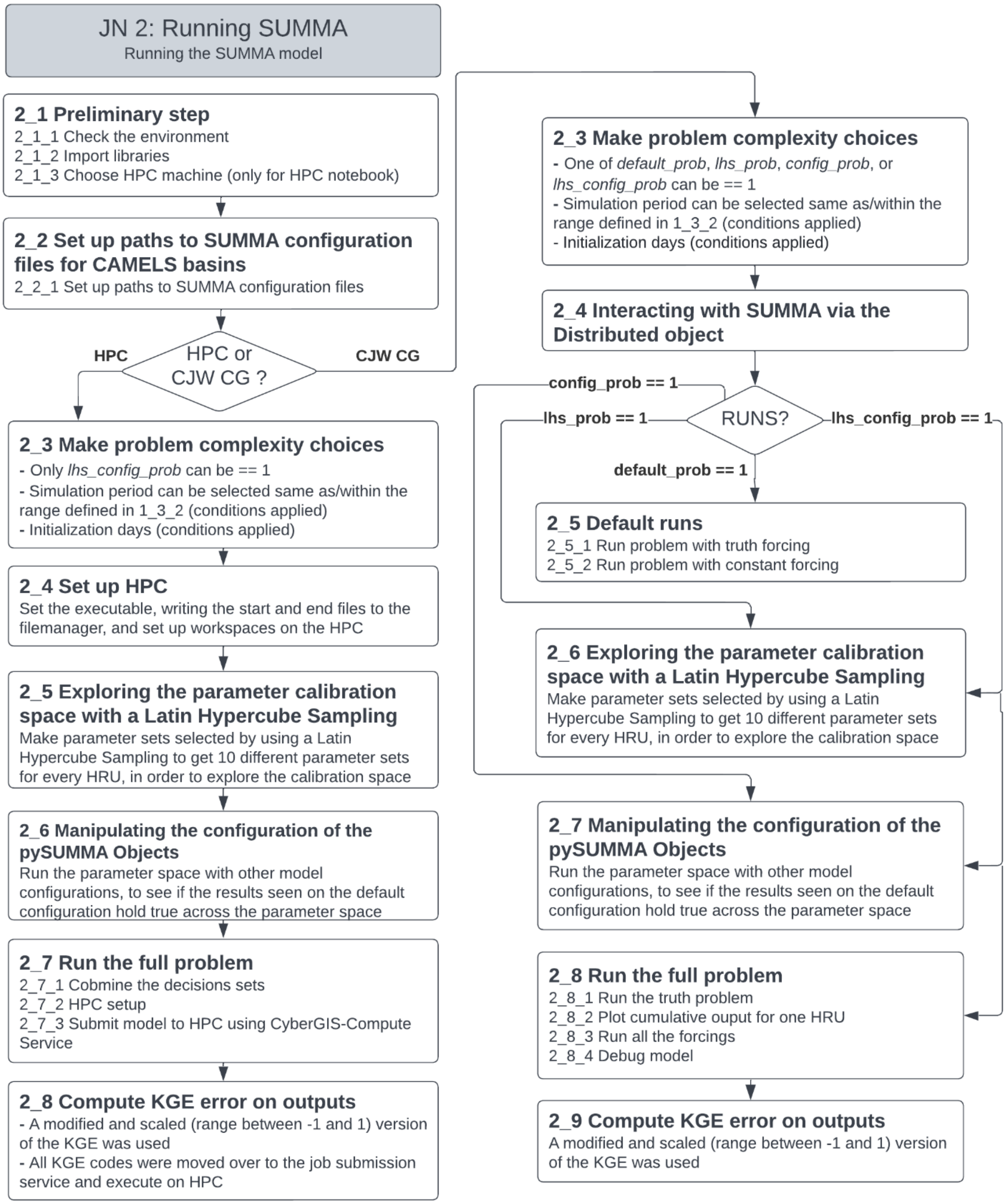

Figure A3. Running SUMMA notebook (JN2) diagram.

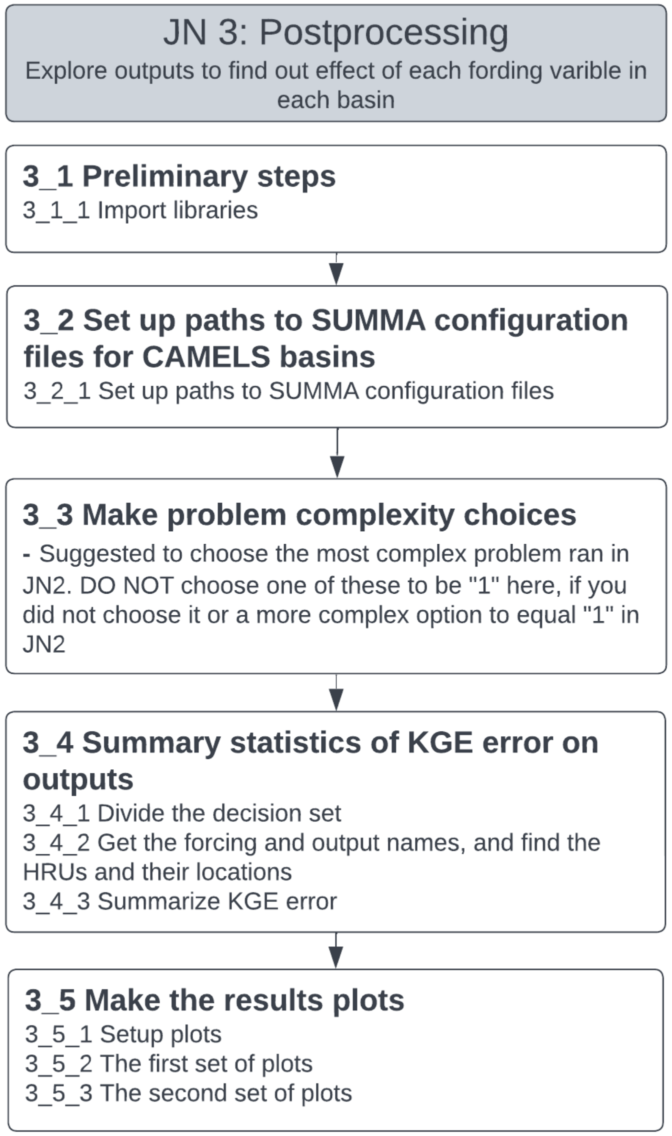

Figure A4. Post-processing notebook (JN3) diagram.

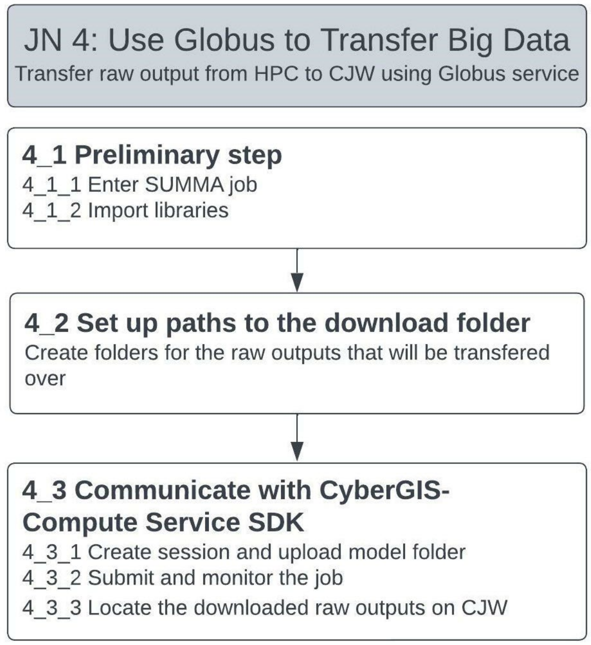

Figure A5. HPC Data transfer notebook (JN4) diagram.

Table A1. SUMMA output variables chosen for analysis (source: Van Beusekom et al., 2022).

#	Variable Type	SUMMA Variable Name	Description (units)
1	liquid water fluxes for the soil domain	SurfaceRunoff	surface runoff (m s-1)
2		AquiferBaseflow	baseflow from the aquifer (m s-1)
3		Infiltration	infiltration of water into the soil profile (m s-1)
4		RainPlusMelt	rain plus melt (m s-1)
5		SoilDrainage	drainage from the bottom of the soil profile (m s-1)
6	turbulent heat transfer	LatHeatTotal	latent heat from the canopy air space to the atmosphere (W m-2)
7		SenHeatTotal	sensible heat from the canopy air space to the atmosphere (W m-2)
8		SnowSublimation	snow sublimation/frost (below canopy or non-vegetated) (kg m-2 s-1)
9	snow	SWE	snow water equivalent (kg m-2)
10	vegetation	CanopyWat	mass of total water on the vegetation canopy (kg m-2)
11	derived	NetRadiation	net radiation (W m-2)
12		TotalET	total evapotranspiration (kg m-2 s-1)
13		TotalRunoff	total runoff (m s-1)
14		TotalSoilWat	total mass of water in the soil (kg m-2)

Table A2. Parameters chosen for Latin Hypercube Sampling (source Van Beusekom et al., 2022).

Parameter Name	Minimum	Maximum	Default	Constraints
k_macropore	1.0d-7	0.1	0.0001	
k_soil	1.0d-7	1.0d-5	variable	
theta_sat	0.3	0.6	variable	> critSoilTranspire; > fieldCapacity; > theta_res
aquiferBaseflowExp	1	10	2.0	
aquiferBaseflowRate	0	0.1	0.1	
qSurfScale	1	100	50	
summerLAI	0.01	10	3	
frozenPrecipMultip	0.5	1.5	1	
heightCanopyTop	0.05	100	variable	> heightCanopyBottom
heightCanopyBottom	0	5	variable	
routingGammaShape	2	3	2.5	
routingGammaScale	1	100000	20000	
albedoRefresh	1	10	1.0	
tempCritRain	272.16	274.16	273.16	
windReductionParam	0	1	0.28	

The eight plots generated by Notebook 3 are described as follows:

1. Location of the selected CAMELS basin.
2. KGE values for each CNST forcing dataset (datasets 2-8; Table A2) by output variable using the DEFAULT model runs. This is a subset of Figure 9A from Van Beusekom et al. (2022) [*].
3. Boxplots depicting the range in the KGE values for each set of model runs (DEFAULT, LHS, CONFIG, and COMPREHENSIVE; Table A1) by output variable. Note, boxplots only appear for the model runs selected in Notebook 2. This is a subset of Figure 9B from Van Beusekom et al. (2022).
4. Boxplots depicting the range in the KGE values for each set of model runs (DEFAULT, LHS, CONFIG, COMPREHENSIVE; Table A1) by CNST forcing dataset (datasets 2-8; Table A2). Note, boxplots only appear for the model runs executed in Notebook 2. This is a subset of Figure 9C from Van Beusekom et al. (2022).
5. Ranks 1 - 7 stacked barplots depicting the relative basin KGE rank counts by CNST forcing dataset (datasets 2-8; Table A2) for the 14 SUMMA output variables. Note, bars on this

plot will only appear if the COMPREHENSIVE basin runs are executed in Notebook 2. This is a subset of Figure 8 from Van Beusekom et al. (2022).

6. Ranks 1 - 7 stacked barplots depicting the relative basin KGE rank counts by CNST forcing dataset (datasets 2-8; Table A2) for the eight SUMMA configurations. Note, the complete figure will only appear if the COMPREHENSIVE basin runs are executed in Notebook 2. A stacked bar for the default configuration (BlL) will be plotted if the LHS basin runs are executed in Notebook 2. This is a subset of Figure 8 from Van Beusekom et al. (2022).
7. Boxplots for each output variable depicting the range in the seven-summed KGE values (from CNST forcing datasets 2-8) for the eight SUMMA configurations, or for the default configuration if only the default configuration was run (DEFAULT or LHS basin runs in Notebook 2. This is a subset of Figure 6 from Van Beusekom et al. (2022).
8. Boxplots depicting the range in the summed SUMMA hourly output variables over the period of record produced using the benchmark (NLDAS) forcing dataset for the eight SUMMA configuration, or for the default configuration if only the default configuration was run (DEFAULT or LHS basin runs in Notebook 2). Note, a point will appear instead of a boxplot if only the default parameter set was run (DEFAULT or CONFIG basin runs in Notebook 2). This analysis is not in Van Beusekom et al. (2022); it is included in the interactive tool to supply users with potential SUMMA output variable ranges for their selected basin.

* To reproduce the modeling case study presented in the current paper, the selected four basins need to be specified in Notebook 1 (Figure A2, “Step 1_3_2 Select basins and simulation period”) and then Notebook 3 can be used to reproduce Figure 3 (KGE values using the DEFAULT model runs for each CNST dataset (datasets 2-8; Table A2), grouped by SUMMA output variable)

References

Bush, R., Dutton, A., Evans, M., Loft, R., Schmidt, G.A., 2021. Perspectives on Data Reproducibility and Replicability in Paleoclimate and Climate Science. Harvard Data Sci. Rev. 2. https://doi.org/https://doi.org/10.1162/99608f92.00cd8f85

Chard, K., Tuecke, S., Foster, I., 2016. Globus: Recent enhancements and future plans, in: Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale. https://doi.org/10.1145/2949550.2949554

Chen, M., Voinov, A., Ames, D.P., Kettner, A.J., Goodall, J., Jakeman, A.J., Barton, M.C., Harpham, Q., Cuddy, S.M., DeLuca, C., Yue, S., Wang, J., Zhang, F., Wen, Y., Lü, G., 2020. Position paper: Open web-distributed integrated geographic modelling and simulation to enable broader participation and applications. Earth-Science Rev. https://doi.org/10.1016/j.earscirev.2020.103223

Choi, Y.-D., Beusekom, A. Van, Li, Z., Nijssen, B., Hay, L., Bennett, A., Tarboton, D., Maghami, I., Goodall, J., Clark, M.P., 2022a. Hydrologic Model Sensitivity to Temporal Disaggregation of Meteorological Forcing Data across CONUS [WWW Document]. HydroShare. URL https://www.hydroshare.org/resource/c0e8de47aee744d088db7019d78c2b3f/

Choi, Y.-D., Beusekom, A. Van, Li, Z., Nijssen, B., Hay, L., Bennett, A., Tarboton, D., Maghami, I., Goodall, J., Clark, M.P., 2022b. SUMMA Simulations using CAMELS Datasets on CyberGIS-Jupyter for Water [WWW Document]. HydroShare. URL https://www.hydroshare.org/resource/50e9716922dc487981b71e2e11f3bb5d/

Choi, Y.-D., Beusekom, A. Van, Li, Z., Nijssen, B., Hay, L., Bennett, A., Tarboton, D., Maghami, I., Goodall, J., Clark, M.P., 2022c. SUMMA Simulations using CAMELS Datasets for HPC use with CyberGIS-Jupyter for Water [WWW Document]. HydroShare. URL https://www.hydroshare.org/resource/9d73d61696ee4f6b9c9a11e21cd44e24/

Choi, Y.-D., Goodall, J., Sadler, J.M., Castronova, A.M., Bennett, A., Li, Z., Nijssen, B., Wang, S., Clark, M.P., Ames, D.P., Horsburgh, J.S., Yi, H., Bandaragoda, C., Seul, M., Hooper, R., Tarboton, D.G., 2021. Toward open and reproducible environmental modeling by integrating online data repositories, computational environments, and model Application Programming Interfaces. Environ. Model. Softw. 135. https://doi.org/10.1016/j.envsoft.2020.104888

Clark, M.P., Nijssen, B., Lundquist, J.D., Kavetski, D., Rupp, D.E., Woods, R.A., Freer, J.E., Gutmann, E.D., Wood, A.W., Brekke, L.D., Arnold, J.R., 2015a. The structure for unifying multiple modeling alternatives (SUMMA), Version 1.0: Technical description.

Clark, M.P., Nijssen, B., Lundquist, J.D., Kavetski, D., Rupp, D.E., Woods, R.A., Freer, J.E., Gutmann, E.D., Wood, A.W., Brekke, L.D., Arnold, J.R., Gochis, D.J., Rasmussen, R.M., 2015b. A unified approach for process-based hydrologic modeling: 1. Modeling concept. Water Resour. Res. 51, 2498–2514. https://doi.org/10.1002/2015WR017198

Clark, M.P., Vogel, R.M., Lamontagne, J.R., Mizukami, N., Knoben, W.J., Tang, G., Gharari, S., Freer, J.E., Whitfield, P.H., Shook, K.R., Papalexiou, S.M., 2021. The abuse of popular performance metrics in hydrologic modeling. Water Resour. Res. 57, p.e2020WR029001. https://doi.org/https://doi.org/10.1029/2020WR029001

Computational and Information Systems Laboratory, 2017. Cheyenne: SGI ICE XA System (NCAR Community Computing). Boulder, CO: National Center for Atmospheric Research; [WWW Document]. https://doi.org/https://doi.org/10.5065/D6RX99HX

CyberGIS-Compute Service, 2021. What is CyberGIS-Compute Service? [WWW Document]. URL https://cybergisxhub.cigi.illinois.edu/knowledge-base/components/cybergis-compute/what-is-cybergis-compute/ (accessed 2.14.23).

CyberGIS Center HydroShare Development Team, 2022. CyberGIS-Jupyter for Water (CJW) Announcements [WWW Document]. URL http://www.hydroshare.org/resource/a901d83a1281404fae58cc41c1cc9889

Essawy, B.T., Goodall, J., Voce, D., Morsy, M.M., Sadler, J.M., Choi, Y.-D., Tarboton, D.G., Malik, T., 2020. A taxonomy for reproducible and replicable research in environmental modelling. Environ. Model. Softw. 134. https://doi.org/10.1016/j.envsoft.2020.104753

Essawy, B.T., Goodall, J.L., Xu, H., Rajasekar, A., Myers, J.D., Kugler, T.A., Billah, M.M., Whitton, M.C., Moore, R.W., 2016. Server-side workflow execution using data grid technology for reproducible analyses of data-intensive hydrologic systems. Earth Sp. Sci. 3, 163–175. https://doi.org/https://doi.org/10.1002/2015EA000139

Expanse System Architecture [WWW Document], 2022. . San Diego Super Comput. Cent. URL https://www.sdsc.edu/services/hpc/expanse/expanse_architecture.html (accessed 5.6.22).

Expanse User Guide [WWW Document], 2022. . San Diego Super Comput. Cent. URL https://www.sdsc.edu/support/user_guides/expanse.html (accessed 5.6.22).

Foster, I., 2011. Globus Online: Accelerating and democratizing science through cloud-based services. IEEE Internet Comput. 15, 70–73. https://doi.org/10.1109/MIC.2011.64

Gan, T., Tarboton, D.G., Dash, P., Gichamo, T.Z., Horsburgh, J.S., 2020. Integrating hydrologic modeling web services with online data sharing to prepare, store, and execute hydrologic models. Environ. Model. Softw. 130. https://doi.org/https://doi.org/10.1016/j.envsoft.2020.104731

Gichamo, T.Z., Sazib, N.S., Tarboton, D.G., Dash, P., 2020. HydroDS: data services in support of physically based, distributed hydrological models. Environ. Model. Softw. 125, 104623. https://doi.org/https://doi.org/10.1016/j.envsoft.2020.104623

Gupta, H.V., Kling, H., Yilmaz, K.K., Martinez, G.F., 2009. Decomposition of the mean squared error and NSE performance criteria: Implications for improving hydrological modelling. J. Hydrol. 377, 80–91. https://doi.org/https://doi.org/10.5194/hess-23-4323-2019.

Hancock, D.Y., Fischer, J., Lowe, J.M., Snapp-Childs, W., Pierce, M., Marru, S., Coulter, J.E., Vaughn, M., Beck, B., Merchant, N., Skidmore, E., Jacobs, G., 2021. Jetstream2: Accelerating cloud computing via Jetstream, in: Practice and Experience in Advanced Research Computing (PEARC '21). Association for Computing Machinery, New York, NY, USA, p. Article 11, 1–8. https://doi.org/https://doi.org/10.1145/3437359.3465565

Horsburgh, J.S., Morsy, M.M., Castronova, A.M., Goodall, J., Gan, T., Yi, H., Stealey, M.J., Tarboton, D.G., 2016. HydroShare: Sharing Diverse Environmental Data Types and Models as Social Objects with Application to the Hydrology Domain. J. Am. Water Resour. Assoc. 52. https://doi.org/10.1111/1752-1688.12363

Hutton, C., Wagener, T., Freer, J., Han, D., Duffy, C., Arheimer, B., 2016. Most computational hydrology is not reproducible, so is it really science? Water Resour. Res. 52, 7548–7555. https://doi.org/10.1002/2016WR019285

Knoben, W.J.M., Clark, M.P., Bales, J., Bennett, A., Gharari, S., Marsh, C.B., Nijssen, B., Pietroniro, A., Spiteri, R.J., Tang, G., Tarboton, D.G., Wood, A.W., 2022. Community Workflows to Advance Reproducibility in Hydrologic Modeling: Separating model-agnostic and model-specific configuration steps in applications of large-domain hydrologic models. Water Resour. Res. https://doi.org/https://doi.org/10.1029/2021WR031753

Kurtz, W., Lapin, A., Schilling, O.S., Tang, Q., Schiller, E., Braun, T., Hunkeler, D., Vereecken, H., Sudicky, E., Kropf, P., Franssen, H.J.H., 2017. Integrating hydrological modelling, data assimilation and cloud computing for real-time management of water resources. Environ.

Model. Softw. 93, 418–435. https://doi.org/https://doi.org/10.1016/j.envsoft.2017.03.011

Kurtzer, G.M., Sochat, V., Bauer, M.W., 2017. Singularity: Scientific containers for mobility of compute. PLoS One 12, e0177459. https://doi.org/10.1371/journal.pone.0177459

Lee, A., 2014. pyDOE [WWW Document]. URL https://pythonhosted.org/pyDOE/

Li, Z., 2021. Dockerfile to create Singularity image for HPC resource [WWW Document]. URL https://github.com/cybergis/cybergis-compute-v2-summa/tree/main/images (accessed 5.11.22).

Li, Z., Michels, A., Lu, F., Padmanabhan, A., Wang, S., 2022. CyberGIS-Jupyter for Water [WWW Document]. HydroShare. URL http://www.hydroshare.org/resource/4cfd280e8eb747169b293aec2862d4f5

Lyu, F., Yin, D., Padmanabhan, A., Choi, Y.-D., Goodall, J.L., Castronova., A.M., Tarboton, D.G., Wang, S., 2019. Reproducible hydrological modeling with CyberGIS-Jupyter: a case study on SUMMA, in: Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (Learning). pp. 1–6. https://doi.org/https://doi.org/10.1145/3332186.3333052

Mathevet, T., Michel, C., Andréassian, V., Perrin, C., 2006. A bounded version of the Nash-Sutcliffe criterion for better model assessment on large sets of basins, in: Large Sample Basin Experiments for Hydrological Model Parameterization: Results of the Model Parameter Experiment–MOPEX. IAHS PUBLICATION, Wallingford, UK, pp. 211–219.

Melsen, L. A., Torfs, P.J.J.F., Uijlenhoet, R., Teuling, A.J., 2017. Comment on “Most computational hydrology is not reproducible, so is it really science?” by Christopher Hutton et al. Water Resour. Res. 53, 2568–2569. https://doi.org/10.1002/2016WR020208

Merkel, D., 2014. Docker: lightweight Linux containers for consistent development and deployment. Linux j 239, no. 2.

Mizukami, N., Wood, A., 2021. NLDAS Forcing NetCDF using CAMELS datasets from 1980 to 2018 [WWW Document]. HydroShare. URL http://www.hydroshare.org/resource/a28685d2dd584fe5885fc368cb76ff2a

Mullendore, G.L., Mayernik, M.S., Schuster, D.C., 2021. Open Science Expectations for Simulation-Based Research. Front. Clim. 3. https://doi.org/https://doi.org/10.3389/fclim.2021.763420

Newman, A.J., Clark, M.P., Craig, J., Nijssen, B., Wood, A.W., Gutmann, E.D., Mizukami, N., Brekke, L., Arnold, J.R., 2015a. Gridded ensemble precipitation and temperature estimates for the contiguous United States. J. Hydrometeorol. 16, 2481–2500. https://doi.org/https://doi.org/10.1175/JHM-D-15-0026.1

Newman, A.J., Clark, M.P., Sampson, K., Wood, A., Hay, L.E., Bock, A., Viger, R.J., Blodgett, D., Brekke, L., Arnold, J.R., Hopson, T., Duan, Q., 2015b. Development of a large-sample watershed-scale hydrometeorological data set for the contiguous USA: Data set characteristics and assessment of regional variability in hydrologic model performance.

Hydrol. Earth Syst. Sci. 19, 209–223. https://doi.org/https://doi.org/10.5194/hess-19-209-2015, 2015

NLDAS-2, 2014. NLDAS-2 Forcing Dataset Information [WWW Document]. URL https://ldas.gsfc.nasa.gov/nldas/v2/forcing (accessed 4.9.21).

OPeNDAP, 2021. OPeNDAP User Guide [WWW Document]. URL https://www.earthdata.nasa.gov/opendap-user-guide (accessed 8.12.21).

Samadi, S., Pourreza-Bilondi, M., Wilson, C.A.M.E., Hitchcock, D.B., 2020. Bayesian Model Averaging With Fixed and Flexible Priors: Theory, Concepts, and Calibration Experiments for Rainfall-Runoff Modeling. J. Adv. Model. Earth Syst. 12. https://doi.org/10.1029/2019MS001924

Simmonds, M., Riley, W.J., Agarwal, D., Chen, X., Cholia, S., Crystal-Ornelas, R., Coon, E., Dwivedi, D., Hendrix, V., Huang, M., Jan, A., 2022. Guidelines for Publicly Archiving Terrestrial Model Data to Enhance Usability, Intercomparison, and Synthesis. Data Sci. J. 21.

Stewart, C.A., Cockerill, T.M., Foster, I., Hancock, D., Merchant, N., Skidmore, E., Stanzione, D., Taylor, J., Tuecke, S., Turner, G., Vaughn, M., Gaffney, N.I., 2015. Jetstream: a self-provisioned, scalable science and engineering cloud environment, in: Proceedings of the 2015 XSEDE Conference: Scientific Advancements Enabled by Enhanced Cyberinfrastructure. St. Louis, Missouri, p. ACM: 2792774. p. 1-8. https://doi.org/https://dx.doi.org/10.1145/2792745.2792774

Tarboton, D., Calloway, C., 2021. THREDDS DAP2 [WWW Document]. URL http://www.hydroshare.org/resource/70070fa1b382496e85ca44894683b15d

Tarboton, D.G., Idaszak, R., Horsburgh, J.S., Ames, D.P., Goodall, J.L., Band, L.E., Merwade, V., Couch, A., Hooper, R.P., Maidment, D.R., Dash, P.K., 2014. HydroShare: Advancing collaboration through hydrologic data and model sharing, in: Proceedings of the 7th International Congress on Environmental Modelling and Software. Int. Environ. Modell. and Software Soc, San Diego, Calif., pp. 978–988.

Towns, J., Cockerill, T., Dahan, M., Foster, I., Gaither, K., Grimshaw, A., Hazlewood, V., Lathrop, S., Lifka, D., Peterson, G.D., Roskies, R., Scott, J.R., Wilkins-Diehr, N., 2014. XSEDE: Accelerating Scientific Discovery. Comput. Sci. Eng. 16, 62–74. https://doi.org/https://dx.doi.org/10.1109/MCSE.2014.80

Van Beusekom, A.E., Hay, L.E., Bennett, A.R., Choi, Y.-D., Clark, M.P., Goodall, J., Li, Z., Maghami, I., Nijssen, B., Wood, A.W., 2022. Hydrologic Model Sensitivity to Temporal Aggregation of Meteorological Forcing Data: A Case Study for the Contiguous United States. J. Hydrometeorol. 23, 167–183. https://doi.org/10.1175/JHM-D-21-0111.1

Virtual Roger User Guide [WWW Document], 2022. . CyberGIS Cent. Adv. Digit. Spat. Stud. - Univ. Illinois Urbana Champaign. URL https://cybergis.illinois.edu/infrastructure/hpc-user-guide/ (accessed 6.5.22).

Yang, C., Raskin, R., Goodchild, M., Gahegan, M., 2010. Geospatial cyberinfrastructure: past, present and future. Computers, Environment and Urban Systems. Comput. Environ. Urban Syst. 34, 264–277. https://doi.org/https://doi.org/10.1016/j.compenvurbsys.2010.04.001

Yin, D., Liu, Y., Padmanabhan, A., Terstriep, J., Rush, J., Wang, S., 2017. A CyberGIS-Jupyter Framework for Geospatial Analytics at Scale, in: Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact. pp. 1–8. https://doi.org/https://doi.org/10.1145/3093338.3093378