Community Workflows to Advance Reproducibility in Hydrologic Modeling: Separating model-agnostic and model-specific configuration steps in applications of large-domain hydrologic models

W. J. M. Knoben[1], M. P. Clark[1,2], J. Bales[3], A. Bennett[4], S. Gharari[5], C. B. Marsh[5], B. Nijssen[6], A. Pietroniro[7], R. J. Spiteri[8], D. G. Tarboton[9], A. W. Wood[10]

[1]Centre for Hydrology, University of Saskatchewan, Canmore, Alberta, Canada
[2]Department of Geography and Planning, University of Saskatchewan, Saskatoon, Saskatchewan, Canada
[3]Consortium of Universities for the Advancement of Hydrologic Science, Inc, Cambridge, Massachusetts, USA
[4]Hydrology & Atmospheric Sciences, University of Arizona, Tucson, Arizona, USA
[5]Centre for Hydrology, University of Saskatchewan, Saskatoon, Saskatchewan, Canada
[6]Civil and Environmental Engineering, University of Washington, Seattle, Washington, USA
[7]Schulich School of Engineering, Department of Civil Engineering, University of Calgary, Calgary, Alberta, Canada
[8]Department of Computer Science, University of Saskatchewan, Saskatoon, Saskatchewan, Canada
[9]Utah Water Research Laboratory, Utah State University, Logan, Utah, USA
[10]National Center for Atmospheric Research, Boulder, Colorado, USA

Key Points:

- Reproducible, transparent modeling increases confidence in model simulations and requires careful tracking of all model configuration steps
- We show an example of globally applicable model configuration code that is traced and shared through a version control system
- Standardizing file formats and sharing of code can increase community-wide efficiency and reproducibility of modeling studies

---

Corresponding author: W. J. M. Knoben, `wouter.knoben@usask.ca`

Abstract

Despite the proliferation of computer-based research on hydrology and water resources, such research is typically poorly reproducible. Published studies have low reproducibility because of both incomplete availability of the digital artifacts of research and a lack of documentation on workflow processes. This leads to a lack of transparency and efficiency because existing code can neither be checked nor re-used. Given the high-level commonalities between existing process-based hydrological models in terms of their input data and required pre-processing steps, more open sharing of code can lead to large efficiency gains for the modeling community. Here we present a model configuration workflow that provides full reproducibility of the resulting model instantiation in a way that separates the model-agnostic preprocessing of specific datasets from the model-specific requirements that specific models impose on their input files. We use this workflow to create both a continental and a local setup of the Structure for Unifying Multiple Modeling Alternatives (SUMMA) framework connected to the mizuRoute routing model. These examples show how a relatively complex model setup over a large domain can be organized in a reproducible and structured way that has the potential to accelerate hydrologic modeling for the community as a whole. We provide a tentative blueprint of how a community modeling paradigm can be built on top of workflows such as this. We term this initiative the "Community Workflows to Advance Reproducibility in Hydrologic Modeling" (CWARHM; pronounced "swarm").

1 Introduction

Confidence in published findings depends on the reproducibility of the study. In much of the recent Earth System sciences research, reproducibility requires knowledge of the computer code and data that underpin a given manuscript. Such code can range from a few lines of code that are used to turn data into figures, or compute certain statistical properties of the data, to modern process-based hydrologic models that can contain many thousands of lines of code. Despite encouraging progress in journal policies (Blöschl et al., 2014; Clark et al., 2021a), it is still difficult to reproduce published findings in the hydrologic sciences (Hutton et al., 2016; Stagge et al., 2019). Stagge et al. (2019) estimate that results may only be reproducible for between 0.6% to 6.8% of nearly 2000 peer-reviewed manuscripts in these fields, because of a lack of sufficiently clearly described methods and a lack of the necessary input data or processing code.

In complex process-based hydrologic model applications, one additional barrier to reproducibility is the effort required to configure the model. It is not uncommon to hear claims that in such modeling studies 80% of overall effort is spent on configuring the model for a specific use case, and only 20% on using the model to answer research questions (e.g. Table 2.8 in Miles, 2014). The effort on model configuration is spent on assembling appropriate data sources for meteorological forcing data and geospatial parameter fields, wrangling these data into the specific format required by the model, defining appropriate model settings, and specifying the required computational infrastructure (e.g., finding the right collection of packages and modules, installing the model, creating the required scripts to run the model. Additional time costs are in dealing with the subjectivity in defining appropriate computational sub-domains (such as where to draw the boundaries for Hydrologic Response Units), interpreting soil and land cover maps, aggregating geospatial data into some form of representative value for a computational unit, and the associated iterative model configuration and testing steps. This process is typically poorly documented and - equally important for reproducibility - extremely time-consuming. In short, the reproducibility problem for process-based hydrologists is made worse by a lack of efficiency in model setup tasks.

Recommended best practices for open, accessible, and reproducible science (e.g. Gil et al., 2016; Hutton et al., 2016; Sandve et al., 2013; Stodden and Miguez, 2013) describe how to improve the reproducibility of computational science. Most focus is currently on advancing the FAIR principles, which state that data, code, and methods must be Findable, Accessible, Interoperable, and Reusable (Wilkinson et al., 2016). Reproducibility requires FAIR data, but also includes sharing details about hardware, software versions, and data versions (Añel, 2017; Bast, 2019; Hut et al., 2017; Sandve et al., 2013). The environmental modeling community is interacting with these prescribed best practices in multiple ways. Choi et al. (2021) identify three ongoing main thrusts aimed at making computational environmental science more open, reusable, and reproducible. First, data and models are increasingly openly available online through services as GitHub, Hydroshare, and institutional repositories. Second, computational environments are increasingly recorded and standardized through container applications such as Docker or in self-documenting notebooks. Third, Application Programming Interfaces (APIs) such as the pySUMMA API from Choi et al. (2021) make interacting with complex models or data increasingly easier.

Comparatively little attention is devoted to efficient reproducibility of the full modeling workflow, which includes data acquisition, data preprocessing, model installation model runs and post-processing of simulations. Efficiency is promoted in a general sense through freely shared code and packages that perform specific tasks in the modeling chain (for example, see Slater et al., 2019, for an overview of R packages that can be used to populate a modeling workflow). However, dedicated efforts to ensure end-to-end reproducibility of modeling studies are less common. Exceptions are Leonard & Duffy (2013, 2014, 2016), who provide an in-depth description of a web-based interface for data preprocessing and visualization of simulations from the PIHM model, geographically constrained to the United States; Havens et al. (2020), who provide an end-to-end workflow for setting up, running, and analyzing a physics-based snow model; and Vorobevskii et al. (2020), who develop an R package that sets up a simple hydrologic model anywhere on the planet for a given domain discretization shapefile provided by the user. Compared to sharing a model's input and output data (which would also enable a study to be reproduced), sharing complete workflows can be more efficient in terms of required storage space. A workflow also provides a transparent record of all modeling decisions, and, more importantly, having a workflow enables a more broadly defined form of reproducibility in which a study can be repeated for a different region, a different data set, or a different version of the same model to see if the original conclusions still hold.

The examples mentioned in the previous paragraph show that it is possible to document workflows for a specific model (or, perhaps more accurately, for a specific version of a model). A further challenge is in designing workflows in such a way that parts of a workflow that configures Model A can be re-used in a workflow that configures Model B. We refer to such a design as separating the model-specific and model-agnostic parts of model configuration (see also Miles and Band, 2015; Miles, 2014, for an example of this concept for geospatial data preparation in the field of ecohydrology). In the case of process-based hydrologic modeling, models such as VIC (Hamman et al., 2018; Liang et al., 1994), MESH (Pietroniro et al., 2007), SUMMA (Clark et al., 2015a, 2015b, 2021c) and SVS (Husain et al., 2016) can be different in how they discretize the modeling domain, the physical processes they include, and the equations used to describe a given process. However, at their core, these models are designed to solve the same general water and energy conversation equations (Clark et al., 2021c). This means that the data requirements for a myriad of extant hydrologic models will vary in the specifics, but are similar in a general sense and include, for example, similar needs for meteorological forcing data and geospatial parameter fields. Preprocessing of these similar data requirements does not need to rely on specifics of the models themselves. For example, in the case of satellite-based MODIS land cover data (details in Appendix A), model-agnostic steps are (1) downloading the source data, (2) stitching the source data together into a coherent global map, (3) projecting this map into the Coordinate Reference System of interest, (4) subsetting from the global data only the domain of interest, and (5) mapping the resulting data in pixels onto model elements. Model-specific steps would be to convert the resulting information (i.e., which pixels/land classes are present per model element) to the specific format a model requires (e.g., the mode of the land class per model element or a distribution of land classes per grid cell), and, if necessary, perform some form of data transformation to connect land class data to model parameter values or settings. Community-wide efficiency gains are possible if workflows distinguish between model-agnostic and model-specific steps and enable straightforward re-use of the workflow for model-agnostic steps (see also Essawy et al., 2016; Gichamo et al., 2020, who makes this argument in the context of web-based model configuration tools).

The hydrologic modeling community is currently not yet at a stage where model configuration code can easily be shared between modeling groups. Model physics code is increasingly distributed under open-source licenses, but the code that creates the necessary model inputs is typically neither well-documented nor available without contacting the model developers. To move towards a culture of community Earth System modeling, we define three distinct elements:

1. For a given model, model configuration code should be publicly available and divided into model-agnostic and model-specific steps;
2. The configuration workflows of multiple different models, ideally using different data sets, should be integrated into a proof-of-concept of a generalized model configuration workflow;
3. A community-wide collaborative effort should refine the proof-of-concept into a flexible model configuration framework.

This paper introduces an open-source model configuration workflow that enables full reproducibility of a process-based hydrologic model setup for any location on the planet. The goals of this workflow and the associated code repository are twofold. Our first goal is to provide an example of open source model configuration code that is divided into model-agnostic and model-specific steps. This advances our long-term goal of fostering a community modeling culture within Earth System sciences, where freely shared model configuration code provides efficiency gains for the modeling community at large. Our second goal is to provide a traceable and efficient basis from which our chosen model configuration can be used. This advances our short-term goal of using this model configuration for a variety of projects by reducing the time commitment needed to create model configurations for different domains and by increasing confidence in the modeling outcomes due to increased transparency and the possibility to reproduce results.

In this paper, we configure models that have traditionally been developed for local or regional use at high spatial and temporal resolutions across large spatial domains using a newly developed workflow that distinguishes between model-agnostic and model-specific configuration steps. In Section 2, we outline several high-level design considerations for model-agnostic workflows. We then use these guidelines to create an example implementation of such a workflow using open-source input data with global coverage, an open-source, spatially distributed, physics-based hydrologic modeling framework designed to isolate individual modeling decisions (Clark et al., 2015a, 2015b, SUMMA; 2021c), and an open-source network routing model (mizuRoute; Mizukami et al., 2021, 2016) to generate hydrologic simulations across multiple spatial scales. Technical details about the models and workflow code are given in Appendix A and B. In Section 3, we present two test cases, covering continental and local-scale model configurations. In Section 4, we reflect on the current state of reproducibility in large-scale hydrologic modeling, with particular focus on why existing efforts have seen only limited uptake and outline a path forward.

2 Increasing efficiency and reproducibility in Earth System modeling

In this section, we propose general guidelines for model configuration workflows in the Earth System sciences. These guidelines are informed by existing efforts to promote reproducibility and efficiency in large-domain modeling efforts, and by our own experience with creating continental domain model configurations for process-based hydrologic models. We consider challenges for novice and advanced modelers. Briefly, our recommendations are as follows:

1. **Separate model-agnostic and model-specific tasks**. The steps in the workflow must remain model-agnostic for as long as possible and provide outputs in commonly used data formats. This increases the potential utility of the code base for use in different projects and for users of different models.
2. **Clarity for modelers**. The workflow must be easily accessible and useable in its default form. A clear structure of the code accompanied by accurate documentation and in-line comments increase the ease-of-use for novice and advanced modelers alike.
3. **Modularity encourages use beyond the original application**. Customization of the workflow must be possible and easy. This makes it possible to adapt, improve, or change specific parts of the workflow to access new data sets, use new processing algorithms, or target different models.
4. **Traceability is key**. Every outcome of each step in the workflow must be accompanied by metadata that describe the configuration code that generated the outcome. This guarantees that, even if changes are made to the model configuration code, any workflow outcome can still be traced back to its original settings.

These considerations inform the general design of our workflow for setting up a large-scale process-based hydrologic and routing model (sections 2.1-2.4). The workflow starts from a user-provided basin mask that defines the area of interest as discrete modeling elements (e.g., grid cells, sub-basins). Such a mask may be derived from digital elevation models (see e.g. TauDEM (Sazib, 2016; Tesfa et al., 2011) or the geospatialtools code base (Chaney and Fisher, 2021)) or obtained from existing basin masks such as HydroBASINS (Lehner and Grill, 2013) or the MERIT Hydro basin delineation (Lin et al., 2019). The scope of our workflow (Figure 1) includes installation of our chosen hydrologic and routing models, pre-processing of forcing and geospatial parameter data, creation of settings files for both models, model runs, and basic visualization of the resulting model simulations.

Our workflow does not extend to the fine-tuning of parameter values through calibration or estimation from auxiliary data sources, which would necessitate synthesizing several decades of research. A wide variety of calibration algorithms exists, each with their own strengths and weaknesses (Arsenault et al., 2014). These are combined with an even wider variety of objective functions that express the (mis)match between a model's simulations and observations of hydrologic states and fluxes (e.g. Murphy, 1988; Clark et al., 2021b; Gupta et al., 2008; McMillan, 2021; Mizukami et al., 2019; Nash and Sutcliffe, 1970; Olden and Poff, 2003; Pushpalatha et al., 2012), using a variety of further choices related to spatial scaling (e.g. Samaniego et al., 2010), regionalization (e.g. Bock et al., 2015) and dimensionality of the calibration problem (e.g. Pechlivanidis et al., 2011). These choices are not easily standardized and require auxiliary data in the form of observations. Inclusion of calibration and parameter estimation methods is therefore planned for future work in an attempt to keep the scope of this first workflow example manageable.

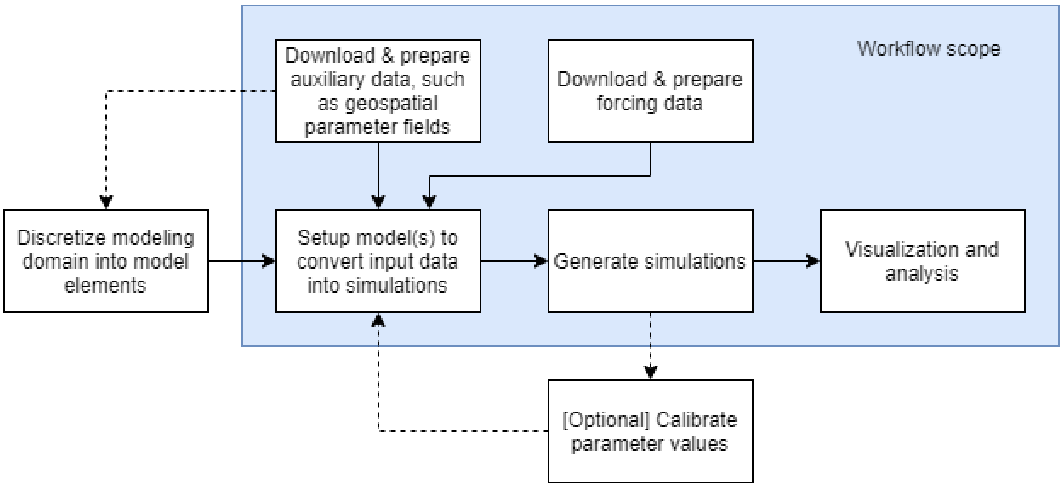

Workflow scope
Download & prepare auxiliary data, such as geospatial parameter fields
Download & prepare forcing data
Discretize modeling domain into model elements
Setup model(s) to convert input data into simulations
Generate simulations
Visualization and analysis
[Optional] Calibrate parameter values

Figure 1. Schematic overview of typical modeling workflow. Scope of the example workflow described in this paper shown in blue. Dashed lines indicate potential connections between elements (such as geospatial parameter fields informing basin discretization) that are not included as part of our workflow.

2.1 Separation of model-agnostic and model-specific tasks

Our first design principle recommends separating model-agnostic and model-specific tasks. Figure 2 shows a high-level overview of model configuration tasks and their connections for the process-based hydrologic model and routing model we use in this paper, and how these can be divided into model-agnostic and model-specific tasks. A detailed overview of model-agnostic and model-specific tasks can be found in Figure 3 and Figure 4 respectively. Despite the seemingly large number of model-specific tasks, time costs (in terms of code development) lean heavily towards the model-agnostic tasks.

Model-agnostic tasks are those tasks that are the same regardless of the model being used, provided that the model requires a given data input (shown in light grey in Figure 2 and Figure 3). In this example, these tasks include the downloading of meteorological forcing data and geospatial parameter fields (digital elevation model (DEM); soil classes and vegetation classes), in some cases clipping raw datasets to the domain of interest (especially if the raw datasets have a global extent) and mapping of these data onto model elements such as grid cells or catchments. Fully model-agnostic outputs in this example are netCDF files of meteorological forcing data (i.e., gridded hourly data at 0.25° latitude/longitude resolution) and GeoTIFF (.tif) files of several geospatial parameter fields.

Model-specific tasks (shown in dark grey in Figure 2 and Figure 4) involve installing the chosen models, transforming the pre-processed data into the specific format the model requires, and running the models. In our workflow implementation, this involves finding the mean elevation, mode land class and mode soil class per model element and exporting the information about the modeling elements (area, latitude and longitude location, slope of the river network, etc.) into the netCDF files our models expect.

Due to the complex nature of existing models and their long histories of development, certain tasks cannot be cleanly separated into model-agnostic and model-specific tasks. The mapping of prepared forcing data and geospatial parameter fields onto model elements (shown in the intermediate grey shade in Figure 2 and Figure 3) is an example of such a task. Certain models run on the same resolution as the forcing/data grid

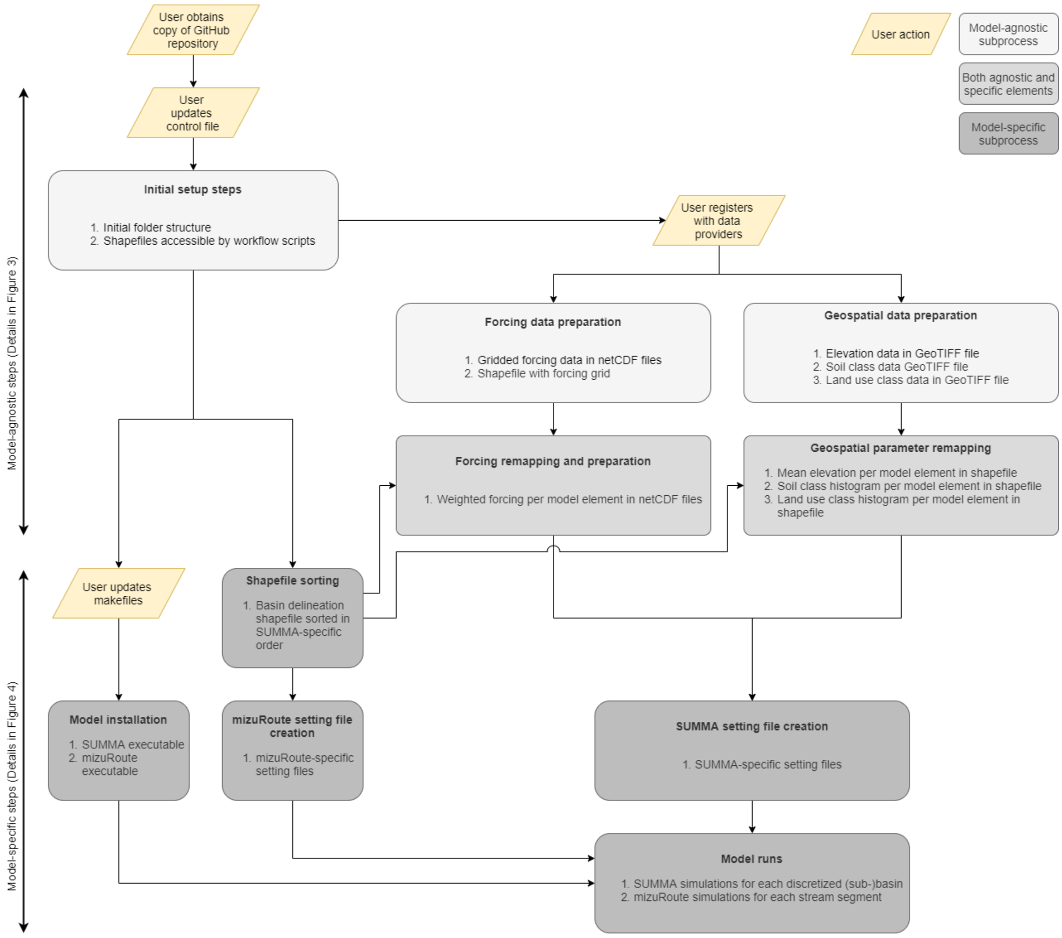

User obtains copy of GitHub repository
User updates control file
Initial setup steps
1. Initial folder structure
2. Shapefiles accessible by workflow scripts
User registers with data providers
Forcing data preparation
1. Gridded forcing data in netCDF files
2. Shapefile with forcing grid
Geospatial data preparation
1. Elevation data in GeoTIFF file
2. Soil class data GeoTIFF file
3. Land use class data in GeoTIFF file
Forcing remapping and preparation
1. Weighted forcing per model element in netCDF files
Geospatial parameter remapping
1. Mean elevation per model element in shapefile
2. Soil class histogram per model element in shapefile
3. Land use class histogram per model element in shapefile
User updates makefiles
Shapefile sorting
1. Basin delineation shapefile sorted in SUMMA-specific order
Model installation
1. SUMMA executable
2. mizuRoute executable
mizuRoute setting file creation
1. mizuRoute-specific setting files
SUMMA setting file creation
1. SUMMA-specific setting files
Model runs
1. SUMMA simulations for each discretized (sub-)basin
2. mizuRoute simulations for each stream segment
Model-agnostic steps (Details in Figure 3)
Model-specific steps (Details in Figure 4)
User action
Model-agnostic subprocess
Both agnostic and specific elements
Model-specific subprocess

Figure 2. High-level overview of model configuration steps, using SUMMA (a process-based hydrologic model) and mizuRoute (a routing model) as example models. Configuration tasks are separated into model-agnostic and model-specific tasks (details in Figure 3 and Figure 4 respectively). Each rounded box specifies the outcomes of that configuration task as a numbered list.

or are able to ingest gridded data in their native alignment and internally map these onto the required model discretization. In our case, this remapping must be done outside the models. In the case of forcing data, the model-agnostic output of meteorological forcing files (named “forcing_yyyymm.nc” in Figure 3) are mapped onto the model elements (catchments in this case), resulting in catchment-averaged model forcing. Temperature time series are further modified with catchment-specific lapse rates to account for elevation differences between the forcing grid and model elements. In the case of parameter fields, intersections between the model-agnostic GeoTIFF files and the shapefile of the modeling domain are generated. These intersections show how often each elevation level, soil class, and land class occurs in each model element. These processes cannot be called truly model-agnostic because some models do not require them, but neither are they fully model-specific. To ensure maximum usability for different models, workflows must be as modular as possible so that modelers can mix and match from available code to suit the particularities of their chosen model (i.e., our third design principle, described later).

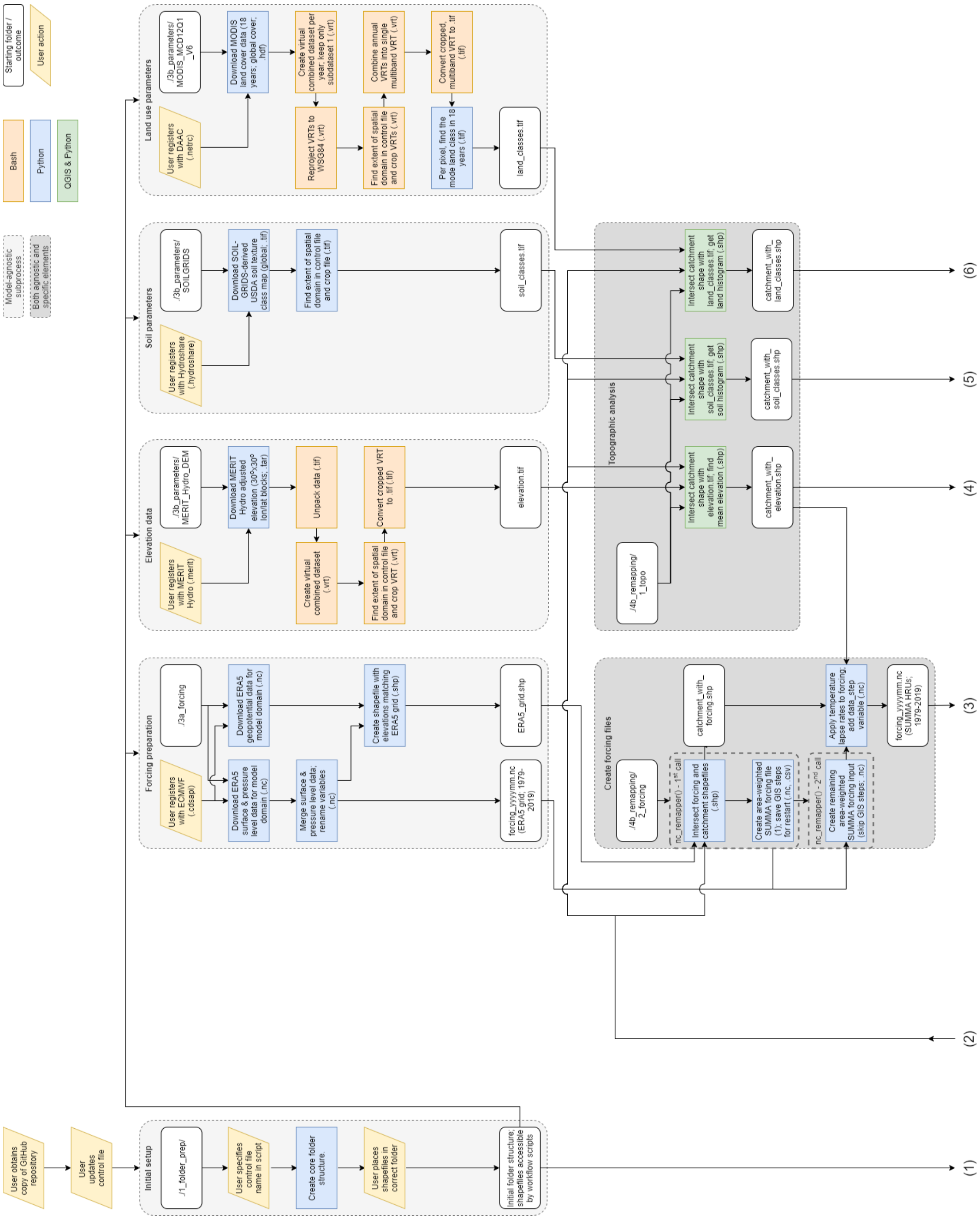

Starting folder / outcome
User action
Bash
Python
QGIS & Python
Model-agnostic subprocess
Both agnostic and specific elements
Land use parameters
/3b_parameters/ MODIS_MCD12Q1_V6
User registers with DAAC (.netrc)
Download MODIS land cover data (18 years; global cover; .hdf)
Create virtual combined dataset per year; keep only subdataset 1 (.vrt)
Reproject VRTs to WSG84 (.vrt)
Combine annual VRTs into single multiband VRT (.vrt)
Find extent of spatial domain in control file and crop VRTs (.vrt)
Convert cropped, multiband VRT to .tif (.tif)
Per pixel, find the mode land class in 18 years (.tif)
land_classes.tif
Soil parameters
/3b_parameters/ SOILGRIDS
User registers with Hydroshare (.hydroshare)
Download SOILGRIDS-derived USDA soil texture class map (global; .tif)
Find extent of spatial domain in control file and crop file (.tif)
soil_classes.tif
Elevation data
/3b_parameters/ MERIT_Hydro_DEM
User registers with MERIT Hydro (.merit)
Download MERIT Hydro adjusted elevation (30°x30° lon/lat blocks; .tar)
Unpack data (.tif)
Create virtual combined dataset (.vrt)
Find extent of spatial domain in control file and crop VRT (.vrt)
Convert cropped VRT to .tif (.tif)
elevation.tif
Forcing preparation
/3a_forcing
User registers with ECMWF (.cdsapi)
Download ERA5 geopotential data for model domain (.nc)
Download ERA5 surface & pressure level data for model domain (.nc)
Merge surface & pressure level data; rename variables (.nc)
Create shapefile with elevations matching ERA5 grid (.shp)
ERA5_grid.shp
forcing_yyyymm.nc (ERA5 grid; 1979-2019)
Topographic analysis
/4b_remapping/ 1_topo
Intersect catchment shape with land_classes.tif; get land histogram (.shp)
catchment_with_land_classes.shp
Intersect catchment shape with soil_classes.tif; get soil histogram (.shp)
catchment_with_soil_classes.shp
Intersect catchment shape with elevation.tif; find mean elevation (.shp)
catchment_with_elevation.shp
Create forcing files
/4b_remapping/ 2_forcing
nc_remapper() - 1st call
Intersect forcing and catchment shapefiles (.shp)
catchment_with_forcing.shp
Create area-weighted SUMMA forcing file (1); save GIS steps for restart (.nc, .csv)
nc_remapper() - 2nd call
Create remaining area-weighted SUMMA forcing input (skip GIS steps; .nc)
Apply temperature lapse rates to forcing; add data_step variable (.nc)
forcing_yyyymm.nc (SUMMA HRUs; 1979-2019)
Initial setup
User obtains copy of GitHub repository
User updates control file
/1_folder_prep/
User specifies control file name in script
Create core folder structure.
User places shapefiles in correct folder
Initial folder structure; shapefiles accessible by workflow scripts
(1)
(2)
(3)
(4)
(5)
(6)

Figure 3. Model-agnostic configuration steps. Each rectangular block corresponds to a specific model setup task and is accompanied by a specific script with Python or Bash code, stored in a GitHub repository. Rounded rectangles indicate starting points of specific sub-tasks (mainly showing which folder in the repository contains certain parts of the workflow) and the outcomes of each sub-process. Parallelograms indicate actions the user must perform. Numbers show connections with the model-specific configuration tasks in Figure 4.

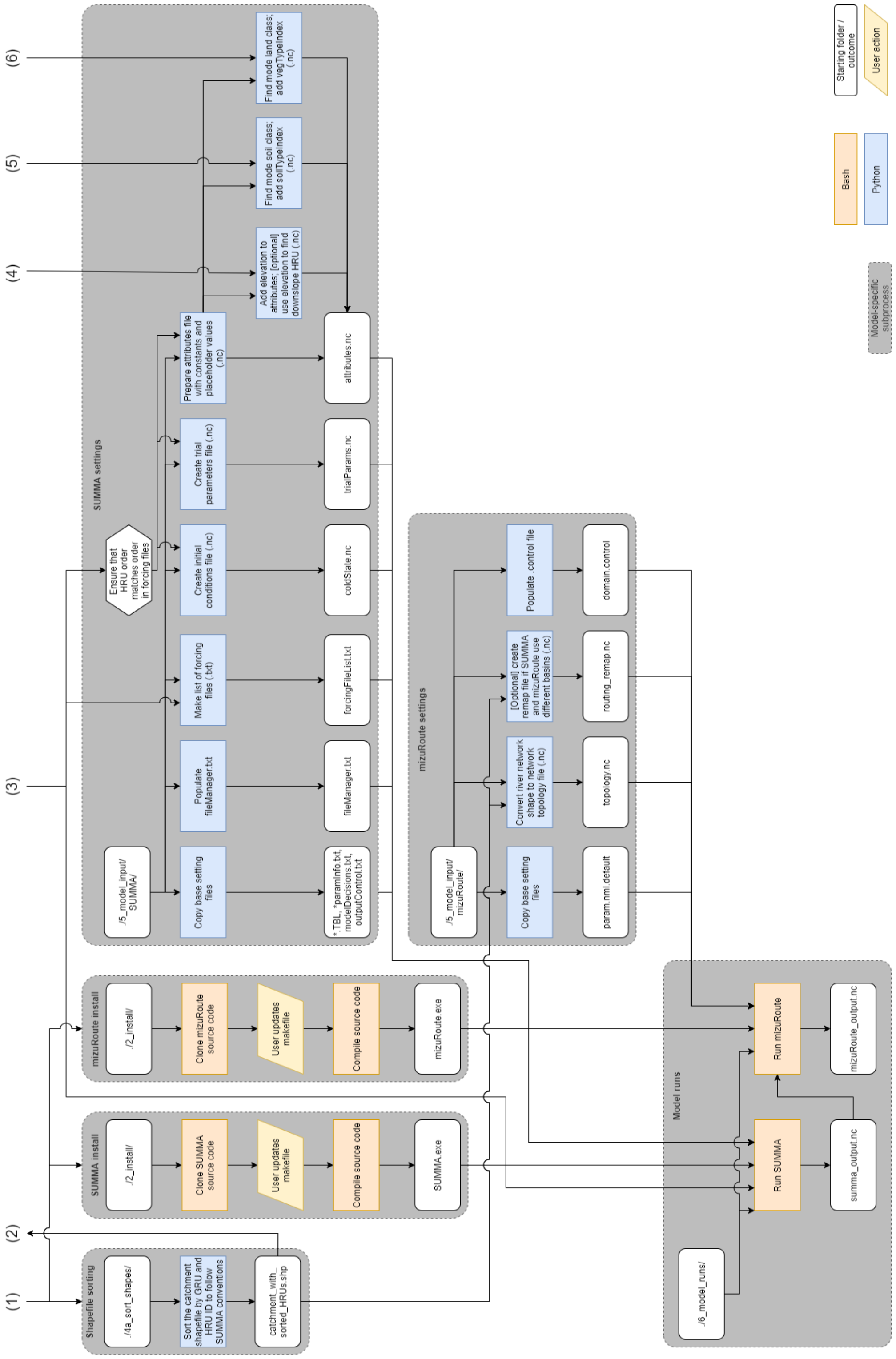

(1)
(2)
(3)
(4)
(5)
(6)
Shapefile sorting
./4a_sort_shapes/
Sort the catchment shapefile by GRU and HRU ID to follow SUMMA conventions
catchment_with_sorted_HRUs.shp
SUMMA install
./2_install/
Clone SUMMA source code
User updates makefile
Compile source code
SUMMA.exe
mizuRoute install
./2_install/
Clone mizuRoute source code
User updates makefile
Compile source code
mizuRoute.exe
SUMMA settings
./5_model_input/SUMMA/
Copy base setting files
Populate fileManager.txt
Make list of forcing files (.txt)
Ensure that HRU order matches order in forcing files
Create initial conditions file (.nc)
Create trial parameters file (.nc)
Prepare attributes file with constants and placeholder values (.nc)
Add elevation to attributes; [optional] use elevation to find downslope HRU (.nc)
Find mode soil class; add soilTypeIndex (.nc)
Find mode land class; add vegTypeIndex (.nc)
*.TBL, *paramInfo.txt, modelDecisions.txt, outputControl.txt
fileManager.txt
forcingFileList.txt
coldState.nc
trialParams.nc
attributes.nc
mizuRoute settings
./5_model_input/mizuRoute/
Copy base setting files
Convert river network shape to network topology file (.nc)
[Optional] create remap file if SUMMA and mizuRoute use different basins (.nc)
Populate .control file
param.nml.default
topology.nc
routing_remap.nc
domain.control
Model runs
./6_model_runs/
Run SUMMA
summa_output.nc
Run mizuRoute
mizuRoute_output.nc
Starting folder / outcome
User action
Bash
Python
Model-specific subprocess

Figure 4. Model-specific configuration steps. Each rectangular block corresponds to a specific model setup task and is accompanied by a specific script with Python or Bash code, stored in a GitHub repository. Rounded rectangles indicate starting points of specific sub-tasks (mainly showing which folder in the repository contains certain parts of the workflow) and the outcomes of each sub-process. Parallelograms indicate actions the user must perform. The hexagon indicates an aspect of SUMMA's input requirements (i.e., not an action or script) and is shown to clarify why creating the forcing files is on the critical path towards creating the other necessary model configuration files. Numbers show connections with the model-agnostic configuration tasks in Figure 3.

2.2 General layout and workflow control

Our second design principle prescribes an intuitive interface for hydrologic modelers. We recognize two elements here: first, the code and data structure must be clear and easy to understand. Second, interacting with the workflow must be straightforward. Our example implementation strives to achieve both of these goals through a clean separation of code and data and the use of a single configuration file (hereafter referred to as a "control file") that outlines high-level workflow decisions such as file paths, spatial and temporal extent of the experiment, and details about the shapefiles that contain the domain discretization. Using configuration or control files is common practice in software design applications (see e.g. Sen Gupta et al., 2015) and avoids the need to introduce hardcoded elements such as file paths and variable values in the code itself.

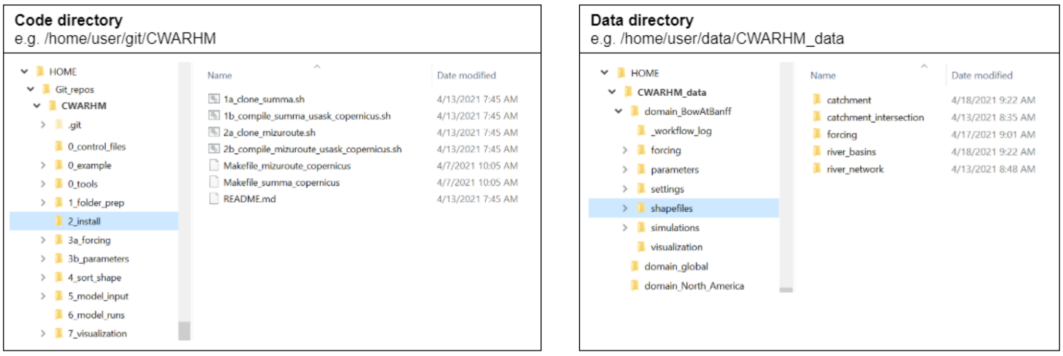
In a typical application of our example workflow, the user first creates a local copy of the code provided on our GitHub repository. We refer to this local code as the "code directory". The workflow is set up to store all data for a given modeling domain in a user-specified data folder (referred to as the "data directory"). This allows a clean separation between the workflow code itself and the data downloaded and preprocessed by the workflow code (Figure 5). The workflow's default settings ensure that the data directory is populated with folders and subfolders with descriptive names, making navigation of the generated data clear.

Figure 6 shows part of the control file that defines the modeling domain for a model configuration for the Bow River at Banff, Canada (the full file is included as part of the repository; see Section 3 for a description of this test case). The control file contains the high-level information needed by the workflow, such as the name of the user's shapefiles, the names of required attributes in each shapefile, the spatial extent of the modeling domain, the years for which forcing data should be downloaded, and file paths and names for all required data. The workflow scripts read information from the control file as needed. Keeping all information in one place enables a user to quickly generate model configurations for multiple domains, without needing to scour all individual scripts for hardcoded file paths, domain extents, etc. For example, changing the simulation period for a given domain requires changing two values in the control file, after which selected code can be re-run to download and preprocess the necessary forcing data and run new simulations. To configure our chosen models for a new domain (assuming that no changes to the model or desired data sets are introduced), a user only needs to provide a new domain discretization file and update in the control file the name of the domain (so that a new data folder can be generated), the names of the discretization files, and the bounding box of the new domain. The workflow can then be fully re-run to create a model configuration for the new domain, without any changes being made to the workflow scripts themselves.

2.3 Flexibility at each step of model setup

Our third design principle recognizes that process-based models are complex entities and that the setup procedures for any given model are model- or even experiment-specific. Not all models will need to go through the same configuration steps, nor will every model experiment need the settings as defined in our example workflow. Our example workflow (Figure 2-4; details in Appendix A) aims to encourage adaptation beyond our original application through modularity and extensive documentation.

First, we have chosen to present the workflow as a collection of scripts rather than a package or module so that the user has straightforward access to the workflow code. This presentation simplifies adapting the code to different models or experiments. Second, the workflow separates model setup into numerous small tasks (Figure 2-4) and saves all intermediate results to files. This modularity makes it straightforward to branch out from our chosen defaults at any given step in the modeling chain. Third, for this iter-



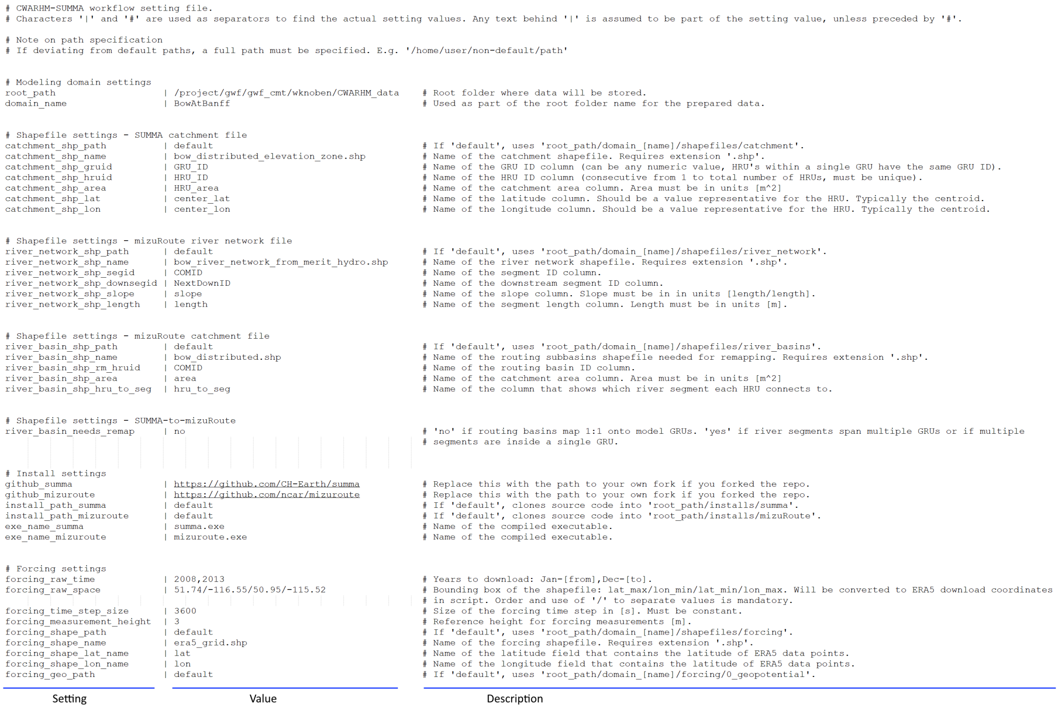
Code directory
e.g. /home/user/git/CWARHM

Name	Date modified
1a_clone_summa.sh	4/13/2021 7:45 AM
1b_compile_summa_usask_copernicus.sh	4/13/2021 7:45 AM
2a_clone_mizuroute.sh	4/13/2021 7:45 AM
2b_compile_mizuroute_usask_copernicus.sh	4/13/2021 7:45 AM
Makefile_mizuroute_copernicus	4/7/2021 10:05 AM
Makefile_summa_copernicus	4/7/2021 10:05 AM
README.md	4/13/2021 7:45 AM

Data directory
e.g. /home/user/data/CWARHM_data

Name	Date modified
catchment	4/18/2021 9:22 AM
catchment_intersection	4/13/2021 8:35 AM
forcing	4/17/2021 9:01 AM
river_basins	4/18/2021 9:22 AM
river_network	4/13/2021 8:48 AM

Figure 5. Example of separated code and data directories. The code directory (left) contains the same data and scripts as available on the repository's GitHub page. The data directory (right) contains the shapefiles, forcing data, parameter data, setting files and model simulations.


Setting
Value
Description

```
# CWARHM-SUMMA workflow setting file.
# Characters '|' and '#' are used as separators to find the actual setting values. Any text behind '|' is assumed to be part of the setting value, unless preceded by '#'.

# Note on path specification
# If deviating from default paths, a full path must be specified. E.g. '/home/user/non-default/path'


# Modeling domain settings
root_path                   | /project/gwf/gwf_cmt/wknoben/CWARHM_data    # Root folder where data will be stored.
domain_name                 | BowAtBanff                                  # Used as part of the root folder name for the prepared data.


# Shapefile settings - SUMMA catchment file
catchment_shp_path          | default                                     # If 'default', uses 'root_path/domain_[name]/shapefiles/catchment'.
catchment_shp_name          | bow_distributed_elevation_zone.shp          # Name of the catchment shapefile. Requires extension '.shp'.
catchment_shp_gruid         | GRU_ID                                      # Name of the GRU ID column (can be any numeric value, HRU's within a single GRU have the same GRU ID).
catchment_shp_hruid         | HRU_ID                                      # Name of the HRU ID column (consecutive from 1 to total number of HRUs, must be unique).
catchment_shp_area          | HRU_area                                    # Name of the catchment area column. Area must be in units [m^2]
catchment_shp_lat           | center_lat                                  # Name of the latitude column. Should be a value representative for the HRU. Typically the centroid.
catchment_shp_lon           | center_lon                                  # Name of the longitude column. Should be a value representative for the HRU. Typically the centroid.


# Shapefile settings - mizuRoute river network file
river_network_shp_path      | default                                     # If 'default', uses 'root_path/domain_[name]/shapefiles/river_network'.
river_network_shp_name      | bow_river_network_from_merit_hydro.shp      # Name of the river network shapefile. Requires extension '.shp'.
river_network_shp_segid     | COMID                                       # Name of the segment ID column.
river_network_shp_downsegid | NextDownID                                  # Name of the downstream segment ID column.
river_network_shp_slope     | slope                                       # Name of the slope column. Slope must be in in units [length/length].
river_network_shp_length    | length                                      # Name of the segment length column. Length must be in units [m].


# Shapefile settings - mizuRoute catchment file
river_basin_shp_path        | default                                     # If 'default', uses 'root_path/domain_[name]/shapefiles/river_basins'.
river_basin_shp_name        | bow_distributed.shp                         # Name of the routing subbasins shapefile needed for remapping. Requires extension '.shp'.
river_basin_shp_rm_hruid    | COMID                                       # Name of the routing basin ID column.
river_basin_shp_area        | area                                        # Name of the catchment area column. Area must be in units [m^2]
river_basin_shp_hru_to_seg  | hru_to_seg                                  # Name of the column that shows which river segment each HRU connects to.


# Shapefile settings - SUMMA-to-mizuRoute
river_basin_needs_remap     | no                                          # 'no' if routing basins map 1:1 onto model GRUs. 'yes' if river segments span multiple GRUs or if multiple
                                                                          # segments are inside a single GRU.


# Install settings
github_summa                | https://github.com/CH-Earth/summa           # Replace this with the path to your own fork if you forked the repo.
github_mizuroute            | https://github.com/ncar/mizuroute           # Replace this with the path to your own fork if you forked the repo.
install_path_summa          | default                                     # If 'default', clones source code into 'root_path/installs/summa'.
install_path_mizuroute      | default                                     # If 'default', clones source code into 'root_path/installs/mizuRoute'.
exe_name_summa              | summa.exe                                   # Name of the compiled executable.
exe_name_mizuroute          | mizuroute.exe                               # Name of the compiled executable.


# Forcing settings
forcing_raw_time            | 2008,2013                                   # Years to download: Jan-[from],Dec-[to].
forcing_raw_space           | 51.74/-116.55/50.95/-115.52                 # Bounding box of the shapefile: lat_max/lon_min/lat_min/lon_max. Will be converted to ERA5 download coordinates
                                                                          # in script. Order and use of '/' to separate values is mandatory.
forcing_time_step_size      | 3600                                        # Size of the forcing time step in [s]. Must be constant.
forcing_measurement_height  | 3                                           # Reference height for forcing measurements [m].
forcing_shape_path          | default                                     # If 'default', uses 'root_path/domain_[name]/shapefiles/forcing'.
forcing_shape_name          | era5_grid.shp                               # Name of the forcing shapefile. Requires extension '.shp'.
forcing_shape_lat_name      | lat                                         # Name of the latitude field that contains the latitude of ERA5 data points.
forcing_shape_lon_name      | lon                                         # Name of the longitude field that contains the latitude of ERA5 data points.
forcing_geo_path            | default                                     # If 'default', uses 'root_path/domain_[name]/forcing/0_geopotential'.
```

Figure 6. Part of a workflow control file, showing settings for the Bow at Banff test case (test cases described in Section 3). The "Setting" column contains specific strings that each script in the repository looks for to identify which line in the control file contains the information the script needs. The "value" column contains the actual settings, such as file paths, shapefile identifiers, etc. Descriptions of each field are included for the user's benefit but not used by the setup scripts.

ation of our workflow, we have chosen to move high-level decisions into the control file and leave various modeling decisions as assumptions in the workflow scripts. We have spent considerable effort on documenting any such assumptions (see Appendix A) to let advanced users make targeted changes to the workflow code. Examples of these decisions include the number of soil layers used across the modeling domain, values for the initial model states, and default routing parameters. In future versions of our workflow, such decisions may be moved to a dedicated experiment-control file.

2.4 Code provenance

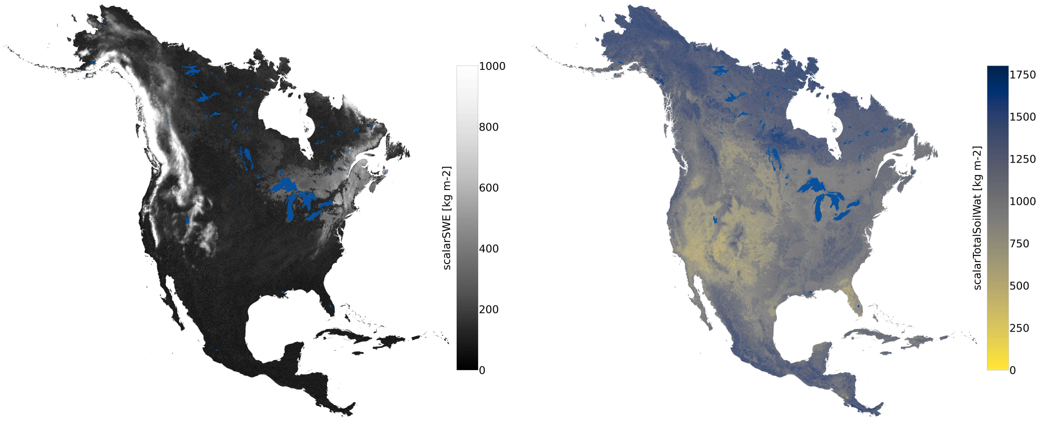
Our fourth design principle relates to traceability. The decision to separate code and data directories potentially introduces a disconnect between code and data, and situations may arise where it is no longer clear which version of a given piece of code generated a particular piece of data. This can happen in cases where the workflow code is updated after having already been used to create (part of) a model configuration. Although the changes to the workflow code can be tracked through version control systems such as Git, it is much more difficult to trace which version of the code generated the data. Every script in our example workflow therefore places both a log file and a copy of its code in the data sub-directory on which it operates. This ensures that, even if a user makes changes to the code directory, a record exists in the data directory of the specific code used to generate the files in that data directory. Copies of the model settings are stored in their simulation data directories by default so that simulation provenance can be traced as well.

3 Test cases

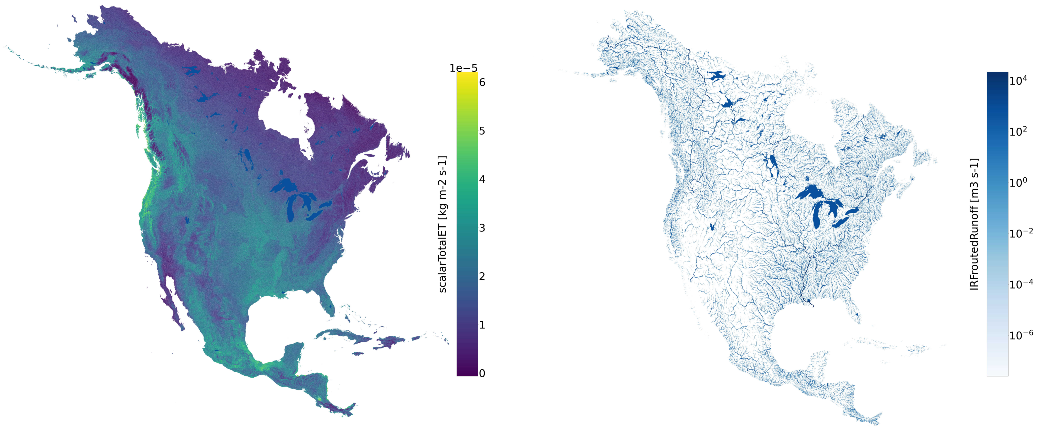
We have implemented a workflow for model configuration using the SUMMA and mizuRoute models. For brevity, details about both models and a detailed description of each step shown in Figure 2-4 can be found in Appendix A. This section briefly describes the two test cases that we used to test our workflow code.

3.1 Continental model instantiation

This first test case uses 40 years of forcing data to simulate hydrologic processes over the North American continent to illustrate the large-domain applicability of our approach. The continental domain is divided into 517,315 sub-basins or Grouped Response Units (GRUs; median GRU size is 33 km^2; mean size is 40 km^2) derived from the global MERIT basins data set (Lin et al., 2019). Simulations are run from 1979-01-01 to 2019-12-31 at a 15-minute temporal resolution to facilitate numerical convergence of the model equations. Figure 7 shows summary statistics of several simulated variables: as expected, snow accumulation tends to be higher in mountainous and higher-latitude locations; total soil water values are lower in the arid regions of the central and western US and Canada and northern Mexico; evapotranspiration rates fluctuate according to available energy (latitude) and water; and large river networks are clearly visible as a result of accumulation of upstream river flow. These results are outputs from a model run with default parameters and are unlikely to be accurate for any given location (see also a cautionary note on the use of global data products in Appendix B). However, the visible large-scale patterns appear hydrologically sensible, giving us confidence that the initial model configuration is a solid basis for further model improvement and development. More importantly, the workflow documents every decision made during model configuration, enabling repeatable simulations of this model domain with only a fraction of the original effort needed.

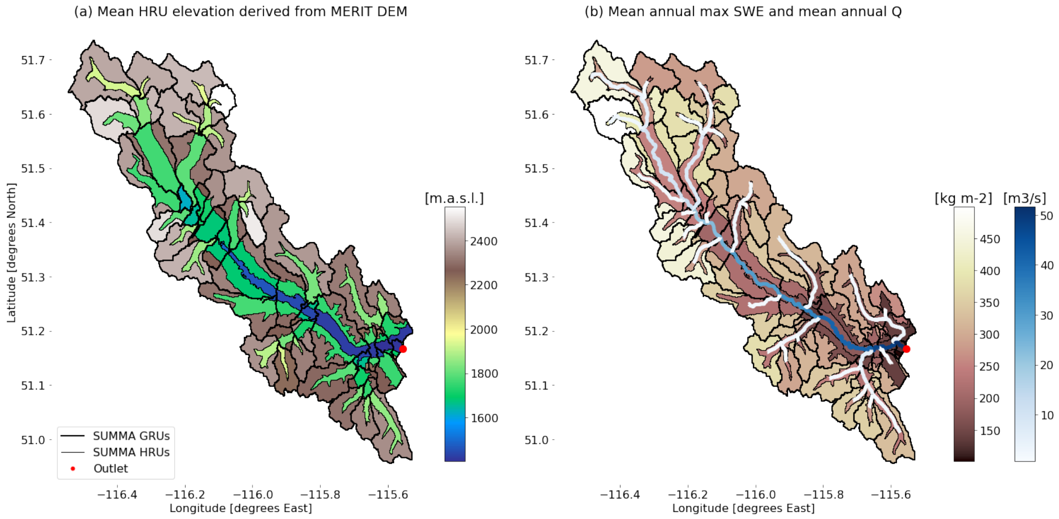

(a) Maximum simulated Snow Water Equivalent

(b) Mean simulated total soil water content


(c) Mean simulated total evapotranspiration

(d) Mean simulated streamflow

Figure 7. Overview of large-domain simulations. SUMMA does not simulate lake hydrology, and no calculations are performed for GRUs that are completely classified as open water. Lake delineations of lakes > 1,000 km^2 are obtained from the HydroLAKES data set (Messager et al., 2016) and used to mask such open-water GRUs in this figure. Model setup uses default parameter values – results for illustrative purposes only. (a) Maximum simulated Snow Water Equivalent per GRU is capped at 1000 [kg m^{-2}] for visualization purposes. (b) Mean simulated total soil water content, which includes both liquid and solid water in the soil profile. (c) Mean simulated evapotranspiration, defined as the sum of evaporation from the soil profile and the canopy, and transpiration by vegetation. Positive values in this image indicate evapotranspiration, while negative values indicate condensation (note that this is the inverse of how evapotranspiration values are stored in the SUMMA output files). (d) Mean runoff as determined by mizuRoute's Impulse Routing Function approach.


(a) Mean HRU elevation derived from MERIT DEM

(b) Mean annual max SWE and mean annual Q

Figure 8. (a) Mean HRU elevation as derived from MERIT Hydro DEM. (b) Mean of maximum SWE per water year shown for each HRU, and mean annual streamflow shown for each river segment. Only data from complete water years is included. Model setup uses default parameter values – results for illustrative purposes only.

3.2 Local model instantiation

A SUMMA Grouped Response Unit (GRU) can be subdivided in as many Hydrologic Response Units (HRUs) as the modeler thinks practical and relevant and these HRUs can be used, for example, to represent different elevation zones, differences in soil or land use, differences in topography, or a combination of several of these elements. As a more localized test case, we created a subset of the MERIT basins data set (Lin et al., 2019) that covers the Bow River (Alberta, Canada) from its source in the Canadian Rocky Mountains to the town of Banff, Canada. We then sub-divided each MERIT sub-basin (GRUs in SUMMA terms) into multiple HRUs based on 500m elevation increments (Figure 8a), created a new control file for this new domain, and re-ran the workflow code. No changes were necessary to any of the workflow scripts because the scripts obtain all the required information from the updated control file and the code is generalized to handle both the large-domain case, where GRUs are not sub-divided into HRUs, and this local case, where HRUs are used.

This test case uses hourly forcing data from 01-01-2008 to 31-12-2013 (again run at a 15-minute sub-step resolution). Temperature lapse rates are applied to the forcing data for each individual HRU (see the "Create forcing files" box in Figure 3), meaning that the hydrometeorological conditions are somewhat different in each HRU despite the forcing grid cells being relatively large compared to the delineated catchments (see Figure 13). Figure 8b shows that simulated Snow Water Equivalent (SWE) varies per HRU and accumulated streamflow varies per stream segment. These figures provide a rudimentary test of the generated model setup (see also a cautionary note on the use of global data products in Appendix B). As may be expected, more snow accumulates at higher elevations, whereas the valley bottoms have a lower snowpack due to warmer air temperatures but larger flows due to their larger accumulated upstream area. As with the continental simulations, this local test case is fully reproducible and all model configuration decisions are stored as part of the workflow. This local test case also shows that different model configurations (in terms of spatial discretization in GRUs and HRUs) can be generated by the same model-specific workflow code.

4 Discussion

4.1 To what extent does our workflow fulfill reproducibility requirements?

Best practices for open and reusable computational science can be briefly summarized as follows (e.g. Gil et al., 2016; Hutton et al., 2016; Stodden and Miguez, 2013): data must be available and accessible, code and methods must be available and accessible, active development on issues with data, code, and methods must be possible, and licensing of data and code should be as permissive as possible. These requirements are formalized in the FAIR principles (Findable, Accessible, Interoperable, Reusable; Wilkinson et al., 2016) but by themselves are not enough to guarantee reproducibility of computational science (e.g., Añel, 2017; Bast, 2019; Hut et al., 2017). To be fully reproducible, details about hardware, software versions, and data versions also need to be recorded and shared (e.g. Choi et al., 2021; Chuah et al., 2020; Essawy et al., 2020). Such practices require a certain time investments but the benefits are clear: the resulting science is more transparent, can be more easily reproduced, and follow-up work can will be more efficient because less time is spent on mundane tasks such as data preparation.

Sandve et al. (2013) outline ten rules for reproducible computational science in the field of Computational Biology, and these are also applicable to Earth System modeling. Our workflow follows nine of these guiding principles:

(1) Our workflow stores copies of the scripts that generate data together with the data itself, which allows a researcher to track how a given result was produced;
(2) Our workflow contains no manual data manipulation: all changes to the data are done in scripts and can be traced;
(3) An exact version of all software used is tracked, partly as installable Python environments and partly on the workflow repository for command line utilities;
(4) All scripts are version controlled on the GitHub repository;
(5) Our workflow is highly modular and stores intermediate results in individual folders to aid in debugging of setups and to allow easy diversion from our workflow;
(6) All data that may support analysis and figures are systematically stored in a logical folder structure;
(7) Our chosen model structure is flexible in prescribing outputs, removing a need to modify the model source code to display specific results;
(8) Our visualization code keeps a precise record of which results file contains the data that underpin a given figure and thus a record exists of which data support a given textual statement about the analysis;
(9) The workflow code is publicly accessible.

Their tenth principle, keeping accurate note of the seeds that underpin any element of randomness in the analysis, does not apply here. Sandve et al. (2013) also recommend sharing access to simulation results. This can be done through repositories such as HydroShare or Pangaea but may be infeasible in the case of large-domain Earth System modeling. For example, storing all input and output data of our continental test case would take approximately 13 TB.

Internal tests on different hardware and by different researchers indicate that our workflow effectively implemented these principles in practice: the workflow can be used to generate identical model inputs and outputs by specifying exact library, package, and model versions. Some caveats apply, however. Although it is possible to trace model source code versions through Git commit IDs, such IDs can obviously not account for local code modifications that are not tracked through Git and good “lab hygiene” is needed to ensure consistency between what is reported to have been done and what has in fact been done. Further, not all data sets that underlie our model setups have Digital Object Identifiers assigned to specific versions of the dataset. Given the size of the data sets involved, sharing the data itself is infeasible, and some care must be taken to precisely track when data were downloaded as a means of making the use of data without DOIs traceable. Last, reproducibility is ensured through specifying exact versions of packages and libraries but many of these packages and libraries are undergoing rapid development and new versions are released constantly. There is a potential issue for reproducibility if older software versions for one reason or another are no longer available (though for fully open-source software this should theoretically not happen). New versions of specific software may however become incorporated into a new version of a workflow if they provide some needed functionality. To ensure backward compatibility, such new workflow versions must therefore also be assigned a new DOI so that any specific workflow version can be tracked and re-used when needed.

4.2 Towards community modeling

4.2.1 Short-term benefits of using workflows

This paper introduces a modular model configuration workflow that separates model-agnostic and model-specific configuration steps. The two main benefits of approaching environmental modeling from this angle are clear: configuring multiple modeling experiments becomes much more efficient, and results are reproducible, because all model configuration decisions can be traced. These benefits address two problems that currently affect Earth System modeling. First, creating a typical model configuration is both difficult and time consuming, and it is possible that model configuration tasks do not receive the attention they deserve. Code may not be checked as thoroughly as may be necessary because bugs may not be readily apparent, and any time spent on model configuration is consequently not spent on writing journal articles or meeting report deadlines. Configuring models can be more efficient if model configuration code is freely and openly shared. This enables time that is currently spent on creating model configurations to instead be spent on in-depth analysis, improving the model representation of real-world processes, and fixing any bugs that may be found in the configuration code or the model source code. If bugs are found, tracing the experiments that are affected by these bugs is possible, and it will be clear which studies need to be corrected. Openly shared model configuration code therefore has the potential to increase the robustness of model simulations and accelerate advances in modeling capabilities. Second, by publishing workflow code alongside a manuscript, the provenance of scientific results remains traceable (see e.g., Hutton et al., 2016; Melsen et al., 2017). This can increase confidence in model results. It also enables more effective follow-up studies because all decisions that underpin the original study can be found in the public domain.

4.2.2 Long-term vision for community workflows

We see workflows such as the one presented in this paper as the first step towards a community-wide modeling framework. Figure 9 illustrates an example of such a framework using the workflow code presented in this paper as examples of each framework layer (see also Miles, 2014; pp 57-58). In addition to a division between model-specific and model-agnostic tasks, we envision a framework that distinguishes between data-specific and data-agnostic preprocessing steps. Processing layers would be separated by standardization layers that prescribe the output format for the preceding processing layer and consequently the input format for the following processing layer. Community-wide agreement on the formats used in standardization layers will promote efficient interoperability of different data-specific processing modules, possibly as part of broader work on international hydrologic standards (see e.g. HY_Features; Blodgett and Dornblut, 2018). Using our workflow as an example, we have created data-specific processing modules for ERA5 meteorological data, SOILGRIDS-derived soil classes, MODIS-derived land classes, and a MERIT Hydro-derived DEM. These modules generate data in standardized formats (in this case, netCDF4 forcing data and GeoTIFF spatial maps) that in turn feed into the data-agnostic

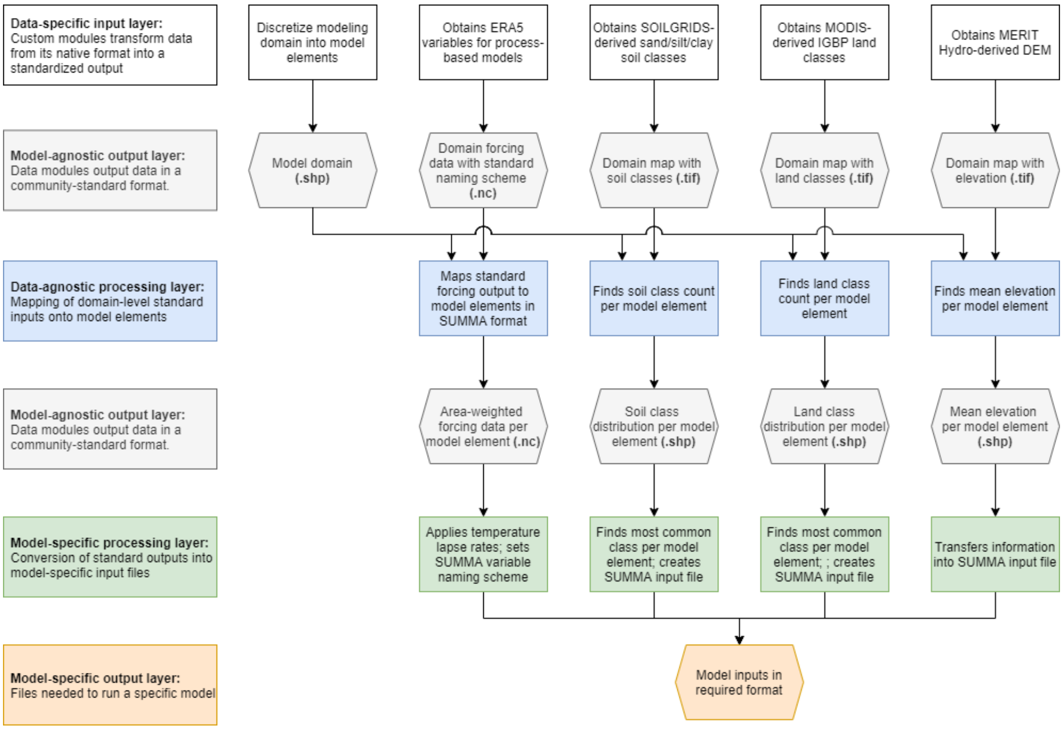

Data-specific input layer: Custom modules transform data from its native format into a standardized output
Model-agnostic output layer: Data modules output data in a community-standard format.
Data-agnostic processing layer: Mapping of domain-level standard inputs onto model elements
Model-agnostic output layer: Data modules output data in a community-standard format.
Model-specific processing layer: Conversion of standard outputs into model-specific input files
Model-specific output layer: Files needed to run a specific model
Discretize modeling domain into model elements
Obtains ERA5 variables for process-based models
Obtains SOILGRIDS-derived sand/silt/clay soil classes
Obtains MODIS-derived IGBP land classes
Obtains MERIT Hydro-derived DEM
Model domain (.shp)
Domain forcing data with standard naming scheme (.nc)
Domain map with soil classes (.tif)
Domain map with land classes (.tif)
Domain map with elevation (.tif)
Maps standard forcing output to model elements in SUMMA format
Finds soil class count per model element
Finds land class count per model element
Finds mean elevation per model element
Area-weighted forcing data per model element (.nc)
Soil class distribution per model element (.shp)
Land class distribution per model element (.shp)
Mean elevation per model element (.shp)
Applies temperature lapse rates; sets SUMMA variable naming scheme
Finds most common class per model element; creates SUMMA input file
Finds most common class per model element; ; creates SUMMA input file
Transfers information into SUMMA input file
Model inputs in required format

Figure 9. Schematic overview of a generalized community modeling framework, populated with examples from our SUMMA setup configuration workflow. Key to this modular approach is community-wide agreement on the formats used in each model-agnostic standardization layer. Such standards enable a modular approach to model-configuration, where existing modules can be seamlessly replaced, as long as they are designed to read and output data in the agreed-upon formats.

remapping layer. This layer generates further model-agnostic data in netCDF4 and Shapefile formats that are then transformed into SUMMA's inputs through a model-specific processing layer.

Our currently defined model-agnostic tasks are of course still implicitly SUMMA-centric (i.e., we have completed those tasks because they generate the data that SUMMA requires), though in principle the outputs can be immediately used by other models. The modular nature of our workflow makes adding new datasets and processing steps as straightforward as writing new data-specific and data-agnostic routines and inserting them in a further unchanged workflow. Changing to a different model requires writing a new model-specific interface layer, but existing data processing scripts can remain untouched (again, assuming that the new model has data needs that can be met by already existing data modules). This means that the workflow can be tailored to a specific model or experiment in a fraction of the time needed to create the model setup from scratch. The modified workflow can then be published alongside the new modeling results to keep those results traceable.

It is of course possible that our attempt to separate model tasks is not equally applicable to every different model that is currently in use. In such cases, we hope that providing a tangible example of how the model configuration code can be organized and shared in a structured way will nevertheless inspire others to create their own workflows. Modifying our workflow or adapting it for different purposes in such ways is the second step we anticipate as needed to move towards a community modeling paradigm. By creating new or modifying existing workflows for new experiments and models, the required structure of a generalized model setup workflow may become apparent. As a third and final step, this generalized workflow can be formalized into a community-driven modeling framework that enhances efficiency and transparency in Earth System modeling.

To initiate the process of creating a community-driven modeling framework, our workflow is available as open-source code: `https://github.com/CH-Earth/CWARHM` (last access: 2021-12-03). We have chosen a permissive license that allows others to freely use and modify our code under the conditions that the modified code base is published under the same license, with attribution of its source and a list of changes. We envision a gradual process in which our repository is modified by others (either piecemeal or by incorporating our entire codebase in a new repository as, for example, a Git submodule), increasingly more data-specific and models-specific processing capabilities are made public, and appropriate formats for standard file formats become apparent. Deciding if and how to integrate these different elements into a single modeling framework is a decision the community will need to make in due course.

4.2.3 Where do workflows stand in the existing reproducibility landscape in hydrologic modeling?

We approach the workflow problem from a catchment modeling perspective within the wider Earth System modeling community (see the definitions of different communities in Archfield et al., 2015). Calls for more efficient, transparent, and shareable model configuration approaches are not new in the catchment modeling community (see e.g., Blair et al., 2019; Famiglietti, et al., 2011; Hutton et al., 2016; Tarboton et al., 2009; Weiler and Beven, 2015) and considerable progress along these lines has been made. For example, Sen Gupta et al. (2015) standardize model inputs and outputs to efficiently couple a snow accumulation and melt routine with an existing open source modeling framework; Miles and Band (2015) develop a Python API that automatically preprocesses ecohydrologic parameter fields and forms the basis of a model configuration workflow for the RHESSys model (Miles, 2014); the Landlab toolkit (Bandaragoda et al., 2019) provides a general interface for building and coupling multiple models; Gan et al. (2020) integrate a web-based hydrologic model service with a data sharing system to promote reproducible workflows; HydroDS (Gichamo et al., 2020) is a web-based service that can be used to prepare input data for modeling; Bennett et al. (2020) create a tool to estimate hourly forcing input for physics-based models from commonly available daily data; Bavay et al. (published online 2020; under review) describe a tool that can be used to effectively create a Graphical User Interface for a given model; Essawy et al. (2020) provide an example of how containerization (storing a full computational environment into a software container) enhances reproducibility; and Kurtzer et al. (2017) develop a means of saving and transferring software and computing environments on and between High Performance Computing clusters. Put together, most if not all elements for fully reproducible, easy-to-use, computational hydrology already exist. So far however, uptake of these tools is regrettably not widespread.

We speculate that uptake of existing tools is somewhat low for multiple reasons. First, these tools are typically provided as self-contained packages. Packages tend to be easy to use for their intended purpose but take time to understand and do not necessarily provide much flexibility to deviate from their intended purpose. Layering additional functions on top of an existing package or modifying a package's source code is certainly possible, but can be outside the comfort zone of many users. Second, several of these tools are provided as web-based services. This can be appealing because, for example, data can be pre-downloaded to speed up model configuration and model simulations can be easily shared. The advantage of such approaches is that they can be combined with some form of server-side data transformations (e.g., subsetting or averaging), which minimizes data transfers. Storing the inputs for and outputs of large-domain simulations can, however, be cumbersome, and keeping pre-downloaded data up-to-date and sufficient for all user needs takes sustained, long-term effort. A further complication is that it is regrettably common that such web-based services require some form of manual interaction with the webpage, limiting opportunities to automate data acquisition tasks. Third, the lack of community agreement on standard data formats means that developers of new tools typically decide to have their tool output data in a format relevant to their own application, which may not be a format that is widely used by others. It is cumbersome for developers to have their tools ingest multiple different data formats and such functionality is therefore somewhat rare. Community-wide agreement on a set of standard data formats, such as proposed in Figure 9, will make it easier for developers to know which data formats their tools must be able to ingest and produce to guarantee seamless interaction with other existing tools.

In short, some of the existing tools may be overdesigned or unsuitable for where the majority of the community currently stands. Such tools are typically designed by a small group of people, using a proof of concept or test case that is directly applicable to the developers' own work. Developers can make educated guesses about how their tool can be made more general beyond their proof of concept, but actually doing so typically relies on the original developers having both the motivation and opportunity to implement functionality for others (e.g., incorporating new data sets or including model-specific layers for other models) or on new developers being willing to first understand the existing package or web-service and then modifying it.

Our approach to provide a tangible example of how to structure model configuration tasks is different. First, our use of scripts instead of packages is likely much more similar to how many models are currently configured, which makes the barrier to trying our approach low. Second, our use of standardization layers that require intermediate files to be in commonly used data formats makes it easy to adapt small parts of our workflow without needing to change any upstream or downstream configuration tasks. Third, there are clear and immediate benefits of adopting a workflow approach of the type proposed in this paper that are unrelated to how widely (or not) this approach is adopted: creating new configurations for the models used in such workflows will be more efficient and the resulting science will stand on a firmer foundation than closed-source results. Should our approach become more widely adopted then the path to a community modeling framework builds itself: as more examples of model configuration workflows become available, our preliminary sketch of a community modeling framework in Figure 9 can be refined or redrawn. How to design, build, and maintain such a community framework can be decided in due course, and appropriate funding may be sought when needed, instead of preemptively securing funding to build modeling infrastructure without knowing how high community uptake of such infrastructure will be or which functionality the community is looking for. Advancing the paradigm of community modeling requires active participation of the community. By providing an example of the first step in the process of designing the structure of a community modeling framework, we hope to encourage others to be part of this initiative.

5 Conclusions

This paper describes a code base that provides a general and extensible solution to configure hydrologic models. Specifically, the paper provides a tool that can be used to create reproducible configurations of the Structure for Unifying Multiple Modeling Alternatives (SUMMA, a process-based hydrologic model) and mizuRoute (a vector-based routing model). We consider this the implementation of a single model in a general framework that separates model-agnostic and model-specific configuration tasks. Such a separation of tasks makes inclusion of new models in this framework relatively straightforward because most of the data pre-processing code can remain unchanged and only model-specific code for the new model needs to be added.

The critical component of this framework is what we refer to as "standardization layers", which prescribe the details of the file formats that must come out of the preceding processing layer and form the input of the following processing layer. By standardizing inputs and outputs, the code that forms the processing layers only needs to concern itself with these prescribed formats. Changing specific processing modules to, for example, pre-process a different data set, perform a different way of mapping data onto model elements, or prepare input files for a different model, can therefore happen in isolation from the remainder of the workflow as long as the new processing code accepts and returns data in the prescribed formats. We show examples of this approach with multi-decadal continental SUMMA and mizuRoute simulations, and with a local SUMMA configuration that uses a more complex spatial discretization than the continental simulations use.

Future work will involve adding model-specific code for multiple additional models and any needed data-specific preprocessing modules. We have termed this initiative "Community Workflows to Advance Reproducibility in Hydrologic Modeling" (CWARHM; "swarm") and we encourage others to be part of this model-agnostic workflow initiative. The configuration code for the SUMMA and mizuRoute setup shown in this manuscript is available on GitHub: `https://github.com/CH-Earth/CWARHM` (last access: 2021-12-03).

Appendix A Workflow description

This section describes in detail our example of a model setup workflow that follows the design principles outlined in Section 2. The workflow code, model code, software requirements, and data are fully open-source to follow the FAIR principles. The workflow is written in Python and Bash, using input data with global coverage, a spatially distributed, physics-based hydrologic modeling framework designed to isolate individual modeling decisions (Clark et al., 2015a, 2015b, 2021c), and a network routing model (Mizukami et al., 2021, 2016) that connects the individual hydrologic model elements through a river network. This example workflow can be used to generate a basic SUMMA and mizuRoute setup anywhere on the globe and is designed such that the model-agnostic parts of the code can easily feed into other modeling chains.

Part of the code in this repository is adapted from or inspired by work performed at the National Centre for Atmospheric Research and the University of Washington.

A1 Models

This section provides a brief overview of SUMMA (Clark et al., 2015a, 2015b, 2021c) and mizuRoute (Mizukami et al., 2021, 2016) to the extent relevant to understand our workflow. We refer the reader to the original papers that describe each model for further details. We selected both models for their flexible nature, computational capacity to model very large domains, and availability of local expertise. Both models are written in Fortran, and their source code needs to be compiled before the models can be used.

A11 Structure for Unifying Multiple Modeling Alternatives (SUMMA)

SUMMA is a process-based modeling framework designed to isolate specific modeling decisions and evaluate competing alternatives for each decision, with the ability to do so across multiple spatial and temporal configurations. SUMMA solves a general set of mass and energy conservation equations (Clark et al., 2015a, 2021c) and includes multiple alternative flux parametrizations (Clark et al., 2015b). It separates the equations that describe the model physics from the numerical methods used to solve these equations, allowing the use of state-of-the-art numerical solving techniques (Clark et al., 2021c).

SUMMA is available as Free and Open Source Software (FOSS) and under active development (see `https://www.github.com/CH-Earth/summa`).

SUMMA organizes model elements into Grouped Response Units (GRUs) that can each be further subdivided into multiple Hydrologic Response Units (HRUs). This enables flexible spatial discretization of modeling domains. For example, point-scale studies are possible by defining the domain as a single GRU that contains exactly one HRU (see e.g., Clark et al., 2015b; GRU area can be an arbitrary value because all fluxes and states are calculated per unit area). It is equally possible to mimic grid-based model setups such as commonly used in land-surface modeling schemes by defining each GRU to be equivalent to a grid cell and optionally using the HRUs to account for sub-grid variability (e.g., mimicking the tiled grid approach of traditional VIC and MESH setups; Liang et al., 1994; Pietroniro et al., 2007). Finally, GRUs can represent the (sub-)catchments of a given river system with HRUs being areas of similar hydrologic behavior within each GRU. Such model configurations can use GRUs and HRUs of irregular shape, which has several advantages over grid-based setups (see e.g., Gharari et al., 2020). Most importantly, such spatial configurations can accurately follow the actual topography of the modeling domain, and this makes model results easier to visualize and interpret. SUMMA is configured with irregularly shaped computational elements in the test cases presented in this paper.

A12 mizuRoute

mizuRoute is a vector-based river routing model specifically designed for large-domain applications such as modeling of hydrologic processes across a continental domain. It organizes the routing domain into Hydrologic Response Units (HRUs; i.e., catchments) and stream segments that meander through the HRUs and provide connections between them (Mizukami et al., 2021, 2016). It can process inputs from hydrologic models with both grid- and vector-based setups and provides different options for channel routing: Kinematic Wave Tracking (KWT) and Impulse Response Function (IRF). For a given stream segment, the IRF method constructs a set of unique Unit Hydrographs (UH) for each upstream segment which is used to route runoff from each upstream reach independently. In other words, the routed runoff in a given stream segment is a simple sum of the UH runoff generated in all upstream segments. The KWT method instead tracks channel runoff as kinematic waves moving through the stream network with their own celerity. mizuRoute is available as FOSS and under active development (see `https://github.com/ESCOMP/mizuRoute`).

A13 Note on definitions

SUMMA distinguishes between Grouped Response Units (GRUs) and Hydrological Response Units (HRUs). SUMMA's main modeling element is the GRU, which can be sub-divided into an arbitrary number of HRUs. SUMMA can handle GRUs and HRUs of any shape (e.g. points, grid cells, catchments) and these terms therefore refer to model elements of arbitrary shape and size. In this workflow, we use mizuRoute to route runoff between SUMMA's GRUs. Somewhat confusingly, mizuRoute refers to all routing basins as HRUs only and does not use the term GRU. As a result, what SUMMA calls GRUs are referred to as HRUs by mizuRoute. For consistency with both sets of documentation, we use their own terminology for model elements where possible. Figure 8 shows a graphical example of the differences in terminology.

A2 Workflow description

This section briefly describes each step shown in the workflow diagram (Figure 2 in the main manuscript). Figures are generated using the test case configured for the Bow River catchment located in Alberta, Canada (see Figure 8 for an overview of this domain).

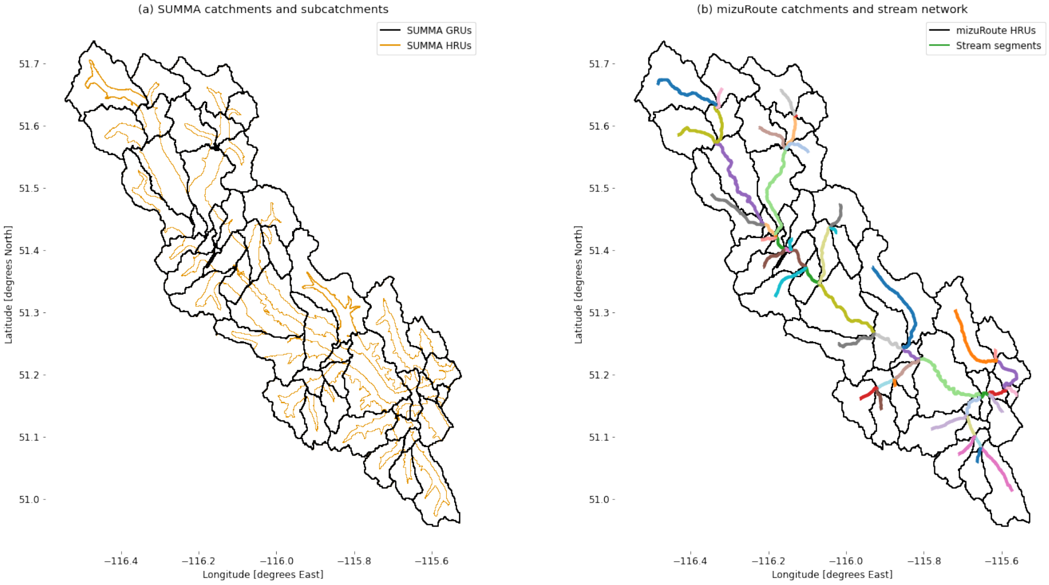


Figure A1. Catchment of the Bow River at Banff (Alberta, Canada) discretized into (a) SUMMA and (b) mizuRoute model elements, showing associated terminology. SUMMA HRUs in (a) represent different elevation bands within each SUMMA GRU. A SUMMA GRU always contains at least one SUMMA HRU. There is no upper limit to the number of HRUs a single SUMMA GRU can be divided into. A single SUMMA HRU is never part of more than one SUMMA GRU. In our example, SUMMA GRUs are identical to mizuRoute HRUs. mizuRoute stream segments are shown in different colors to emphasize that in this case each mizuRoute HRU maps 1:1 onto a single stream segment.

This test case covers a geographically small area (approximately 2200 km^2) and uses a more complex model setup (SUMMA GRUs subdivided into multiple HRUs) than the continental test case (where SUMMA GRUs contain exactly one HRU each), making it the best choice to visualize model setup procedures. Italicized phrases in this section indicate folders, scripts, or variables as found in the GitHub repository. To start, a user would download or clone the complete GitHub repository. The following sections provide more detail about the scripts found within the GitHub repository. A summary of this section and a direct mapping of repository directories onto manuscript sections can be found in the Supporting Information. Although our workflow requires only limited user interaction to generate a model configuration for a new domain, we do make certain assumptions about this model configuration which users should be aware. These assumptions are specified in each subsection.

A21 Workflow setup and folder structure

This section describes the steps "User updates control file" and the steps contained in the box "Initial setup" (Figure 2).

A21.1 Control files Control files are the main way for a user to interact with the workflow. They contain high-level information such as file paths, file names, variable names, and specification of the spatial and temporal extent of the modeling domain (see also Sen Gupta et al., 2015). A new control file needs to be created by the user for each new domain. As an example, the control file for the *Bow_at_Banff* test case is included as part of the Github repository, in the folder *./CWARHM/0_control_files*. The READMEs of each sub-folder on the GitHub repository contain a list of the settings in the control file on which the scripts in that sub-folder rely.

A21.2 Folder preparation The workflow separates generated data from the code used to generate the data. The script in the folder *./CWARHM/1_folder_prep* generates a basic data folder structure in a location of the user's choosing (see Figure 3b). This basic folder structure generates a main data folder with a subdirectory for the current domain. In this domain folder, it further generates a dedicated folder where the user can place their shapefiles that delineate the SUMMA catchments (hydrologic model GRUs and HRUs), mizuRoute catchments (routing model HRUs), and mizuRoute river network. This is the only script in the workflow that needs to be manually modified if a setup for a new domain is generated. A user will need to modify the variable *sourceFile* so that it points to the control file for the current domain. In our example, this is set to *control_Bow_at_Banff.txt*. The script then copies the contents of this control file into a new file called *control_active.txt*, which is the file every other workflow script will search for. The variable *sourceFile* needs to be updated when a control file for a new domain is used. Note that the contents of the file *control_active.txt* determine which folders and files the other workflow scripts operate on.

A21.3 Domain shapefiles With a basic folder structure in place, the user can now move their prepared shapefiles into the newly generated folders (assuming the control file uses 'default' values for these shapefile paths). Briefly, the shapefiles should contain: geometries that delineate the hydrologic model GRUs and HRUs, the routing model HRUs, and the routing model river network in a regular latitude/longitude projection (in other words, in the Coordinate Reference System defined by EPSG:4326; `https://epsg.io/4326` [last access, 2021-10-11]). Each shapefile needs to specify certain properties of the model domain, such as identifiers for each GRU, HRU, and stream segment; HRU area and centroid location, stream segment slope and length; and the stream segment ID into which a given HRU drains.

Detailed requirements for the shapefiles are provided in the README in *./CWARHM/1_folder_prep.* Example shapefiles for the *Bow_at_Banff* test case are part of the repository and can be found in the subfolders of *./CWARHM/0_example.*

A22 Model-agnostic workflow elements

This section provides details about the model-agnostic elements of the workflow (shown in light grey in Figure 2). For convenience, this section is organized to follow the four model-agnostic sub processes: pre-processing of forcing, elevation, soil, and land use data.

A22.1 Pre-processing of forcing data Our chosen forcing product is the ERA5 reanalysis data set (Copernicus Climate Change Service (C3S), 2017; Hersbach et al., 2020) provided by the European Centre for Medium-Range Weather Forecasts (ECMWF). ERA5 data are available as hourly data for the period 1979 to present minus 5 days, at a 31 km spatial grid that covers the Earth's surface or at a re-gridded 0.25° x 0.25° latitude/longitude resolution. ERA5 data preparation includes two-way interactions between atmosphere, land surface and ocean surface components. The ERA5 model setup includes different atmospheric layers and ERA5 data are available at 137 different pressure levels (i.e., heights above the surface), as well as at the surface. The lowest atmospheric level is L137, at geopotential and geometric altitude 10 m (i.e., 10 m above the land surface). To limit the influence of ECMWF's land model on our required forcing variables (simulating the land surface response is SUMMA's role after all), we obtain air temperature, wind speed, and specific humidity at the lowest pressure level (L137; Hersbach et al., 2017) instead of at the land surface. Precipitation, downward shortwave radiation, downward longwave radiation, and air pressure are unaffected by the land model coupling and can be downloaded at the surface level (Hersbach et al., 2018).

Surface and pressure level data are stored in two different data archives and are accessed in different ways. Download scripts for each separate archive are found in folder *./CWARHM/3a_forcing/1a_download_forcing.* These scripts access the C3S Climate Data Store (CDS) using the user's credentials (instructions on how to obtain and store credentials can be found in the README in the download folder) and download the necessary data in monthly blocks of hourly data at a regular 0.25 x 0.25° latitude/longitude resolution. The spatial and temporal extents of the domain are taken from the control file. As per the ERA5 documentation, ERA5 data should be seen as point data, even though standard visualization approaches typically show this kind of data as an interpolated grid. In our example workflow, we make the simple assumption that each ERA5 point contains forcing data that are representative for the grid of size 0.25° x 0.25° of which the grid point is the centroid. The workflow code automatically finds which ERA5 grid points to download based on the catchment bounding box specified in the control file (Figure 9). Once downloaded, the code in *./CWARHM/3a_forcing/2_merge_forcing* can be used to merge the surface and pressure level downloads into a single netCDF file, which is used for further processing. During this merging process, the ERA5 variable names are also changed to more descriptive ones.

Gridded forcing data does not map directly onto irregular model elements such as HRUs. Code in *./CWARHM/3a_forcing/3_create_shapefile* generates a shapefile for the forcing data that outlines the forcing grid (dotted red lines in Figure 9), which is later used to find the relative contribution of each forcing grid cell to the forcing of each HRU. The elevation of each ERA5 grid point is added to this shapefile. This information is later used to apply temperature lapse rates based on the difference in elevation of the ERA5 data and mean HRU elevations. As per the ERA5 documentation, the elevation of each ERA5 data point is found by dividing the geopotential [m^2 s^{-2}] of each point (downloaded through scripts in *./CWARHM/3a_forcing/1b_download_geopotential*) by the gravitational acceleration [m s^{-2}].

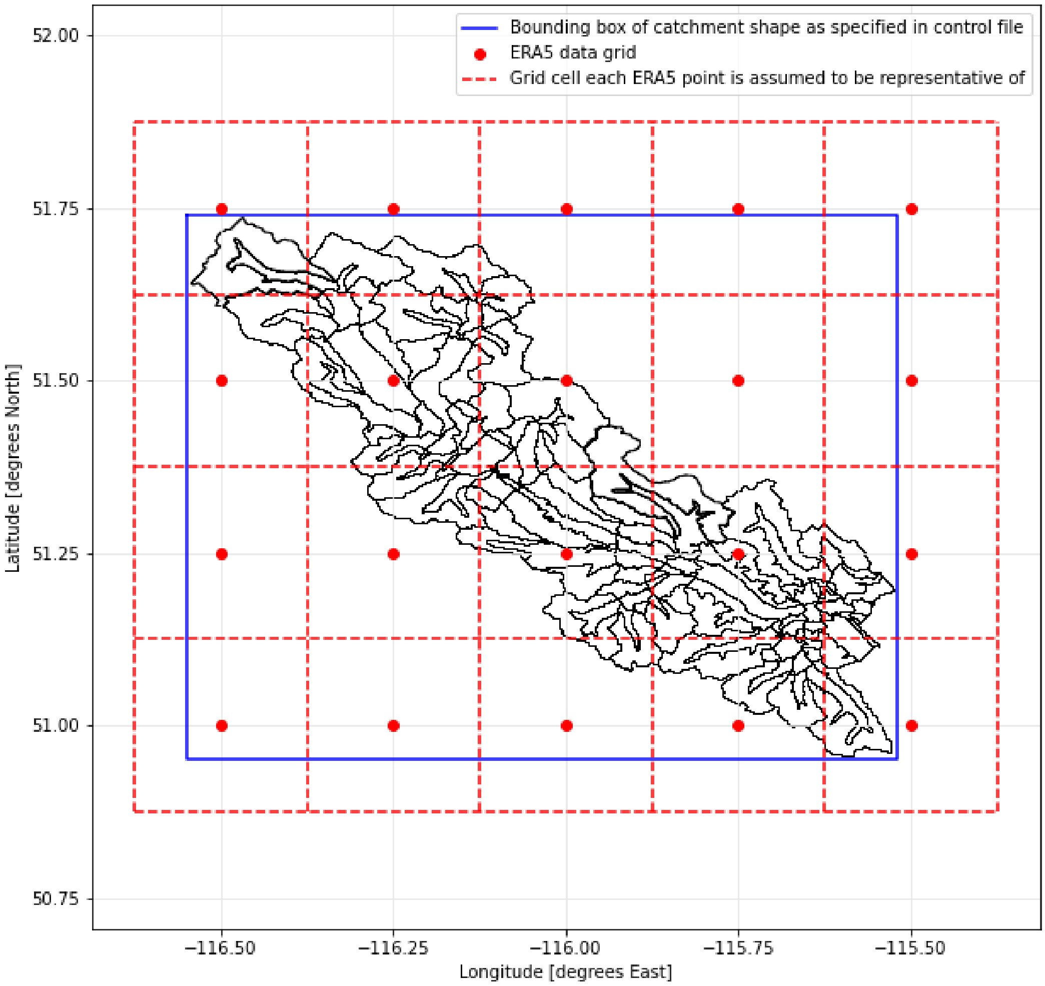


Figure A2. Overview of ERA5 data points, catchment and bounding box and how ERA5 data is assumed to overlap the catchment for the Bow at Banff test case.

Key assumptions in this part of the workflow are (1) that the user has access to the Copernicus Data Store. Instructions on how to obtain access are given in the README in folder *./CWARHM/3a_forcing.* (2) We consider that using forcing data that are the result of interaction between the atmospheric and land surface model components is undesirable and hence somewhat limit this interaction by downloading certain variables at the lowest pressure level instead. (3) ERA5 data points are assumed to be representative of grids of size 0.25° x 0.25°. (4) Gravitational acceleration is assumed to be constant at g = 9.80665 [m s^{-2}] (Tiesinga et al., 2019), although in reality this value would vary depending on latitude and altitude. (5) ERA5 variable names are changed to more descriptive ones that are also the names SUMMA expects these variables to have.

A22.2 Pre-processing of geospatial parameter fields Pre-processing of geospatial parameter fields

Three different types of geospatial data are required for our example model setup. A Digital Elevation Model (DEM) provides the elevation of each HRU and is both a SUMMA input and required to apply temperature lapse rates as a preprocessing step. Maps of soil classes and vegetation classes are needed to utilize parameter lookup tables. These tables specify values for multiple parameters for a variety of soil and land classes. By knowing the soil or land class for a given HRU, SUMMA uses the predefined parameter values for those classes.

Digital Elevation Model We use the hydrologically adjusted elevations that are part of the Merit Hydro dataset (Yamazaki et al., 2019) to determine HRU elevations. The Merit Hydro hydrography maps cover the area between 90° North and 60° South at a spatial resolution of 3-arc seconds. They are derived from the Merit DEM (Yamazaki et al., 2017), which itself is the result of extensive error correction of the SRTM3 (Farr et al., 2007) and AW3D-30m (Tadono et al., 2016) DEMs. Scripts can be found in the subdirectories of *./CWARHM/3b_parameters/MERIT_Hydro_DEM.*

Merit Hydro data are provided as compressed data packages that cover 30° x 30° areas. Based on the spatial extent of the domain, as given in the control file, the required 30° areas are downloaded in compressed format. Data are then uncompressed so that the individual GeoTIFF files are accessible. These files are first combined into a Virtual Dataset (VRT), from which the exact modeling domain is extracted into a new VRT. The VRT with the extracted subdomain is then converted into a single GeoTIFF file that contains the DEM for the modeling domain. A key assumption is that the user has access to the MERIT Hydro data. Instructions on how to obtain access are given in the README in folder *./CWARHM/3b_parameters/MERIT_Hydro_DEM.*

Vegetation classes We use MODIS MCD12Q1_V6 data (Friedl and Sulla-Menashe, 2019) to determine land cover classes at the HRU level. MODIS MCD12Q1 data are available for the years 2001 to 2018 at a 500 m resolution. The data set contains land cover classes for multiple different land cover classification schemes. Each data layer is the result of supervised classification of MODIS reflectance data (Sulla-Menashe and Friedl, 2018). Scripts can be found in the subdirectories of *./CWARHM/3b_parameters/MODIS_MCD12Q1_V6.*

MODIS MCD12Q1 data is provided as multiple Hierarchical Data Format (HDF) files that each cover a part of the planet's surface at a given time. The source data files are in a sinusoidal projection and of irregular shape which makes it difficult to extract a specific region. Therefore, the workflow downloads all available individual HDF files for each year (i.e., global coverage). The individual files for each data year are combined into one Virtual Dataset per year (VRT) for easier processing. Only the data layer of interest, the International Geosphere Biosphere Programme (IGBP) land cover classification, is included in the VRT. The VRT is reprojected from its original sinusoidal projection into a regular latitude/longitude grid (EPSG:4326) from which the modeling domain is extracted. The annual VRTs are then combined into a single multi-band VRT, which is then converted to a multi-band GeoTIFF file. The MODIS documentation advises against using the data of an individual year due to data uncertainty (Sulla-Menashe and Friedl, 2018). Therefore, the mode land class between 2001 and 2018 is identified as the most likely class for each pixel and stored as a new GeoTIFF file.

Key assumptions are (1) that the user has access to NASA's Earth Data website. Instructions on how to obtain access are given in the README in folder *./CWARHM/3b_parameters/MODIS_M* (2) Our example uses the IGBP land cover classification data, which is one of multiple options available.

Soil classes Our example uses a global map of soil texture classes (Knoben, 2021) derived from the SoilGrids 250m dataset (Hengl et al., 2017) to specify representative soil classes at the HRU level. The SoilGrids data are provided at a 250 m resolution and at seven standard depths (up to 2 m depth). Data are the result of a combination of approximately 150,000 observed soil profiles, 158 remote sensing-based soil covariates, and multiple machine learning methods. SoilGrids maps of sand, silt, and clay percentages were converted to a soil texture map for each depth using the soil texture class boundaries of Benham et al. (2009). For each 250 m map point, the mode soil class of the seven soil layers was selected as a representative value for the soil column as a whole, resulting in a single global map of soil texture classes. The pre-processing code needed to create this map (data download, data merge into a coherent map, conversion from percentages to soil texture, finding the mode of each soil column) is accessible as part of the data resource (Knoben, 2021). Scripts can be found in the subdirectories of *./CWARHM/3b_parameters/SOILGRIDS.*

The global soil texture class map is provided at the same horizontal resolution as the underlying SoilGrids data. The workflow first downloads a map with global coverage. The spatial extent of the modeling domain is extracted based on the bounding box specified in the control file and stored as a new GeoTIFF file.

Key assumptions are (1) that the user has access to Hydroshare. Instructions on how to obtain access are given in the README in folder *./CWARHM/3b_parameters/SOILGRIDS and* (2) the global soil map used assumes that mode soil class in each soil column can be considered as the representative soil class for the entire soil column and that the soil properties (such as saturated conductivity and pore volume) for the mode class are representative of the properties of the column. This approach ignores the existence of layered soil profiles and the differences in water movement this can cause (e.g., Vanderborght et al., 2005). This also assumes that the most common class contains the layers that are most hydrologically active and relevant for modeling purposes.

A23 Mapping of data to model elements

This section provides details about the mapping of preprocessed forcing data onto model elements (shown in the intermediate grey shade in Figure 2). This process cannot be called truly model-agnostic because whether it is needed depends on the model in question: some models are able to ingest the pre-processed data directly.

A23.1 Geospatial parameter fields In our example, geospatial data in the form of GeoTIFF files containing the DEM, land classes, and vegetation classes cannot be ingested by the hydrologic model directly. The data must be mapped onto the model elements (HRUs) as delineated in the catchment's shapefile. These procedures use the open-source QGIS project (QGIS Development Team, 2021) to provide the necessary Python functions (*./CWARHM/4b_remapping/1_topo*). Key assumptions are (1) that MERIT Hydrologically Adjusted Elevation data need to be aggregated into mean elevation values per model element, whereas (2) soil and vegetation classes need to be aggregated into histograms that summarize the distribution of values per model element.

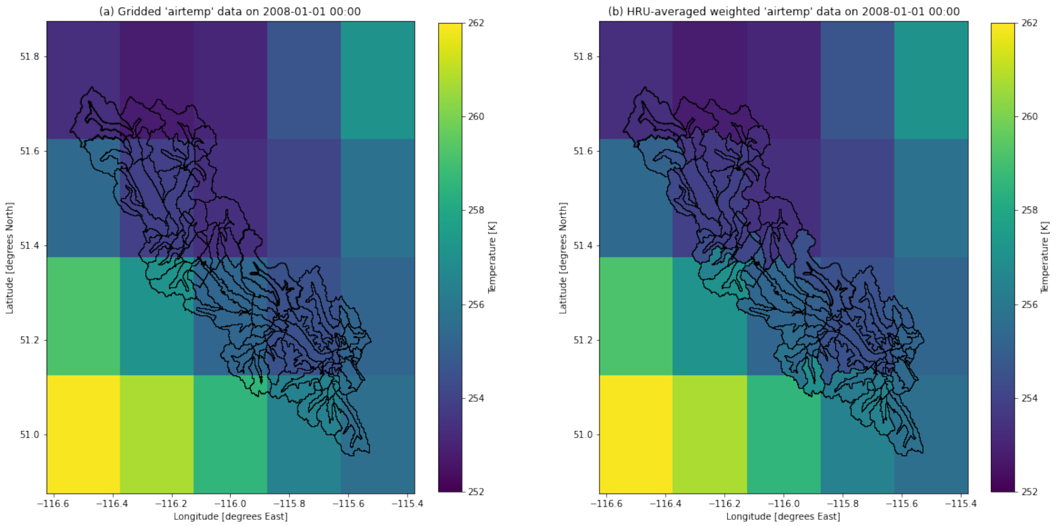


Figure A3. (a) Original gridded air temperature data as found in the ERA5 data. (b) HRU-averaged air temperature obtained as a weighted average of the relative contributions of each ERA5 grid cell to each HRU. Temperatures shown outside the catchment boundaries are the original gridded values.

A23.2 Forcing data Figure 10 shows the original gridded air temperature values on an arbitrary day and the HRU-averaged values on that same day that are obtained by mapping the gridded forcing data onto the model elements. For each model element, the relative overlap with each ERA5 grid cell determines the weight with which that forcing grid cell contributes to the HRU-averaged value. This procedure is applied to all seven forcing variables and all time steps to generate HRU-averaged forcing (*./CWARHM/4b_remapping/2_forcing*).

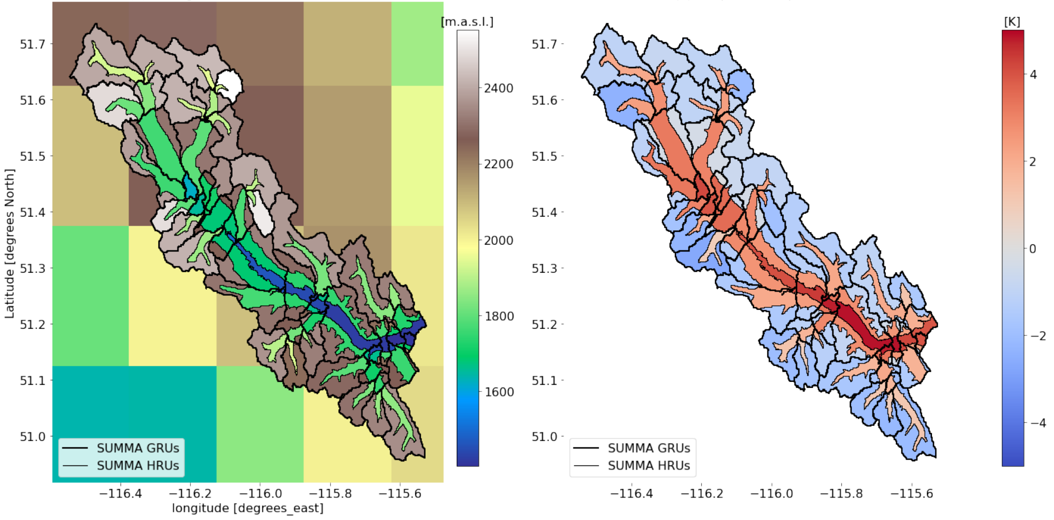
We then apply a constant environmental lapse rate of 0.0065 K m^{-1} (Wallace and Hobbs, 2006, p. 421) to the HRU-averaged air temperature data to account for any differences between ERA5 data point elevation and mean HRU elevation (Figure 11). To avoid excessive data access, the SUMMA-specific variable *data_step* (which specifies the temporal resolution of the forcing data in [s]) is added to each forcing file at the same time as lapse rates are applied.

Key assumptions are that (1) a temporally and spatially constant lapse rate can be used. This is common in gridded analysis but typically not locally accurate (Minder et al., 2010). Local lapse rates may be very different from this assumed value, especially in complex terrain and at seasonal or hourly time scales (Cullen and Marshall, 2011; Minder et al., 2010). Regionally and temporally variable lapse rates are a possible way to improve this part of the workflow (e.g., Dutra et al., 2020)but doing so is beyond the scope of this study. (2) The influence of slope and aspect on radiation fluxes is currently not accounted for in forcing data preparation.

A24 Model-specific workflow elements

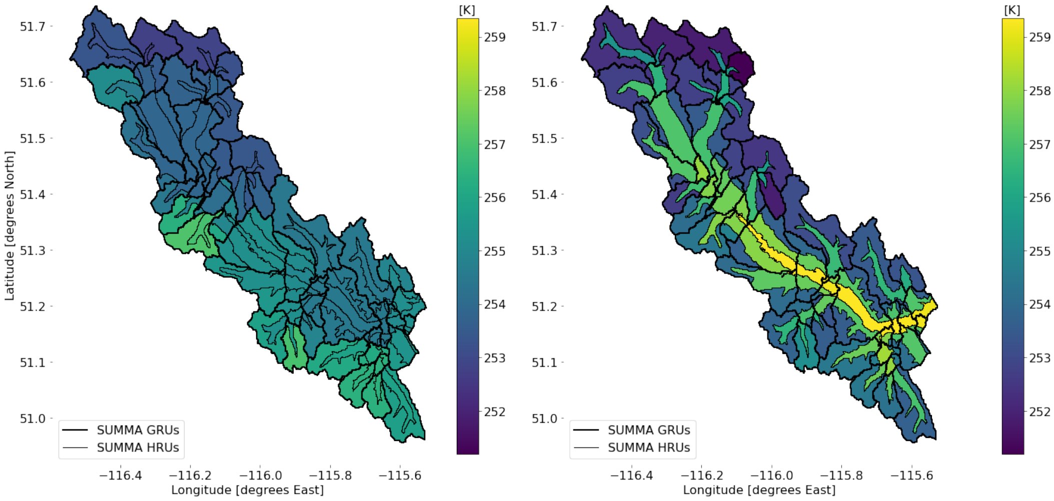
This section provides details model-specific steps of the workflow (shown in dark grey in Figure 2). These steps form the interface between preprocessed data and models.

A24.1 SUMMA and mizuRoute installation The source code for both SUMMA and mizuRoute can be obtained through GitHub (see Section 8.1.1 and 8.1.2). Scripts



(a) ERA5 grid and mean HRU elevation

(b) Temperature lapse rates



(c) HRU-averaged temperature on 2008-01-01 00:00

(d) HRU-averaged temperature with lapse rates on 2008-01-01 00:00

Figure A4. (a) HRU-averaged elevation derived from Merit Hydro adjusted elevations data. ERA5 grid point elevation calculated from geopotential data and a spatially constant gravitational acceleration value, visualized as grid cells. (b) Temperature lapse values based on a constant lapse rate and a weighted difference between ERA5 grid point elevation and HRU mean elevation. (c) Air temperature data before lapse rates are applied. (d) Air temperature data after lapse rates are applied.

in *./CWARHM/2_install* provide code to download the latest version of both models to a local machine. Both models are written in Fortran and need to be compiled to create executables. The exact commands and settings needed will vary between different computational environments. The workflow contains examples of model compile code for a specific High Performance Computing environment.

Key assumptions are as follows. (1) The user has determined the appropriate settings to compile both models on their own computational infrastructure and made the necessary changes to our provided example code. (2) Both scripts assume that the "develop" branch of each model is the version of interest. (3) A Linux environment is needed for full functionality. A basic alternative that avoids compiling the source code is to install pySUMMA and mizuRoute through Conda, but this provides pre-compiled executables only. Access to the source code is not possible and updates present on GitHub may not immediately appear in the pySUMMA Conda distribution.

A24.2 Shapefile sorting to ensure expected order of model elements SUMMA makes certain assumptions about GRU and HRU order in its input files. These are: (1) GRUs and HRUs are in the same order if the forcing files and all SUMMA input files that contain information at the GRU and HRU level; and (2) HRUs inside a given GRU are found at subsequent indices in each NetCDF file. Note that these requirements do not specify anything about the values of the GRU and HRU IDs and only focus on the order in which the IDs appear in files. The code in *./CWARHM/4a_sort_shape* sorts the shapefile that contains the catchment delineation into GRUs and HRUs before this shapefile is used by other scripts. This is more efficient than postponing this sorting until the SUMMA input files are generated. A key assumption is that computational efficiency is an important consideration and therefore this model-specific requirement should be run before the (model-agnostic) remapping is performed.

A24.3 SUMMA input files SUMMA requires several different configuration files: 1) default parameter values at the GRU and HRU level, 2) *lookup tables* with predefined soil and vegetation parameters for different soil and land classes, 3) a *model decisions* file that specifies which modeling decisions (e.g., the type of numerical solver) and flux parametrizations to use, 4) an *output control* file that specifies which internal model variables to write as model output, at which temporal resolution to do so and which, if any, summary statistics to provide, 5) a *file manager that* specifies the file paths to all model inputs and outputs as well as the time period for the simulation, 6) a *forcing file list* that specifies the names of all meteorological forcing files to use, 7) a *trial parameter* file that can be used to overrule any parameter value specified in the default parameter files and in the lookup tables that can be helpful to quickly test different parameter values during e.g., calibration, 8) an *initial conditions* file that specifies the model states at the beginning of the first time step, and 9) an *attribute file* that contains topographic information such as elevation, soil type, and land use type at the HRU level.

In our example setup, files with default parameter values, lookup tables, model decisions, and requested outputs are provided as part of the repository. These files do not require any information from the preprocessing steps for forcing data and geospatial parameter fields and can therefore simply be copied into the new SUMMA settings directory. The file manager and forcing file list are populated with information available in the workflow control file. The workflow generates a trial parameter file that, for our test cases, specifies a required value for only one parameter. This parameter controls the time resolution of SUMMA's simulations and is here specified as 900 seconds (i.e., four times smaller than the 1-hourly forcing data resolution) to improve numerical convergence of the model equations. The initial conditions file serves a dual purpose: it specifies the model states at the start of the simulation and the vertical discretization of the soil domain into discrete layers. In this example, SUMMA is initialized with eight soil layers of increasing thickness (0.025 m for the top layer, 1.50 m for the bottom layer), without any snow or ice present, with some soil and groundwater liquid water storage and at a constant temperature of the soil, and canopy domains of 10°C. The attributes file is populated with data from the user's shapefiles (GRU and HRU IDs, HRU-to-GRU mapping, longitude and latitude, HRU area) and from the geospatial preprocessing steps. Figure 12 shows the original geospatial parameter fields that are the outcomes of our model-agnostic preprocessing steps and how these are converted into model-specific values for SUMMA's attributes file. All scripts are available in the subdirectories of *./CWARHM/5_model_input/SUMMA*.

Key assumptions are (1) that the HRU and GRU default parameter files, model decisions and lookup table files are assumed to be sensible choices for the domain of interest. In particular, the choice of ROSETTA lookup table for soil properties (NCAR Research Applications Laboratory, RAL, 2021; U.S. Department of Agriculture: Agricultural Research Service (USDA ARS), 2021) and the modified IGBP table for vegetation properties (NCAR Research Applications Laboratory, RAL, 2021) inform how the geospatial data is preprocessed (i.e. which geospatial data sets are used and how they are transformed). (2) Vertical discretization of domain is currently set at eight soil layers with increasing thickness with depth. (3) Initial conditions are dry and warm, and there is no snow and ice present in the domain. (3) Model decisions relying on *contourLength* and *tan_slope* attributes are not supported (currently this is the baseflow model decision *qbaseTopmodel*, as well as certain radiation calculations that account for slope inclination). Attribute variable *downHRUindex* is only used if decision *qbaseTopmodel* is active and is therefore set to zero.

A24.4 mizuRoute input files mizuRoute requires several configuration files: 1) a *default parameter* file that has values for its different routing schemes, 2) a *network topology* file that contains a description of the river network and its properties, 3) optionally, a *remapping file* that shows how output from a hydrologic model should be mapped onto mizuRoute's routing network, and 4) a *mizuRoute.control file* that specifies the necessary file paths and routing settings. In our example setup, a default routing parameter file is provided as part of the repository. This file does not require any information from the preprocessing steps for forcing data and geospatial parameter fields and can therefore simply be copied into the new mizuRoute settings directory. The network topology file contains a description of the routing basins and their associated stream segments. It specifies which basins and segments exist, which segment each basin drains into and physical properties of the domain such as drainage area, segment length and segment slope. The optional remapping file only needs to be used in cases where the hydrologic model operates on model elements that do not map directly onto mizuRoute's routing basins. In such a case the remapping file specifies the weight with which each hydrologic catchment contributes flow to each routing basin. The mizuRoute.control file is populated with information available in the workflow control file. All scripts are available in the subdirectories of *./CWARHM/5_model_input/mizuRoute*.

Key assumptions are (1) that the provided routing parameter values are appropriate for the domain and (2) Hillslope routing (i.e. routing between different SUMMA HRUs inside a given SUMMA GRU) is performed by SUMMA. mizuRoute is configured to do the river network routing between different SUMMA GRUs.

A24.5 Model runs Model runs use the compiled SUMMA and mizuRoute executables to perform simulations using the inputs and settings defined in their respective configuration files (*./CWARHM/6_model_runs*). As part of the model run scripts, model configuration files are copied into the simulations output directories. This ensures traceability of the simulations by keeping a record of the settings used to generate the simulations.

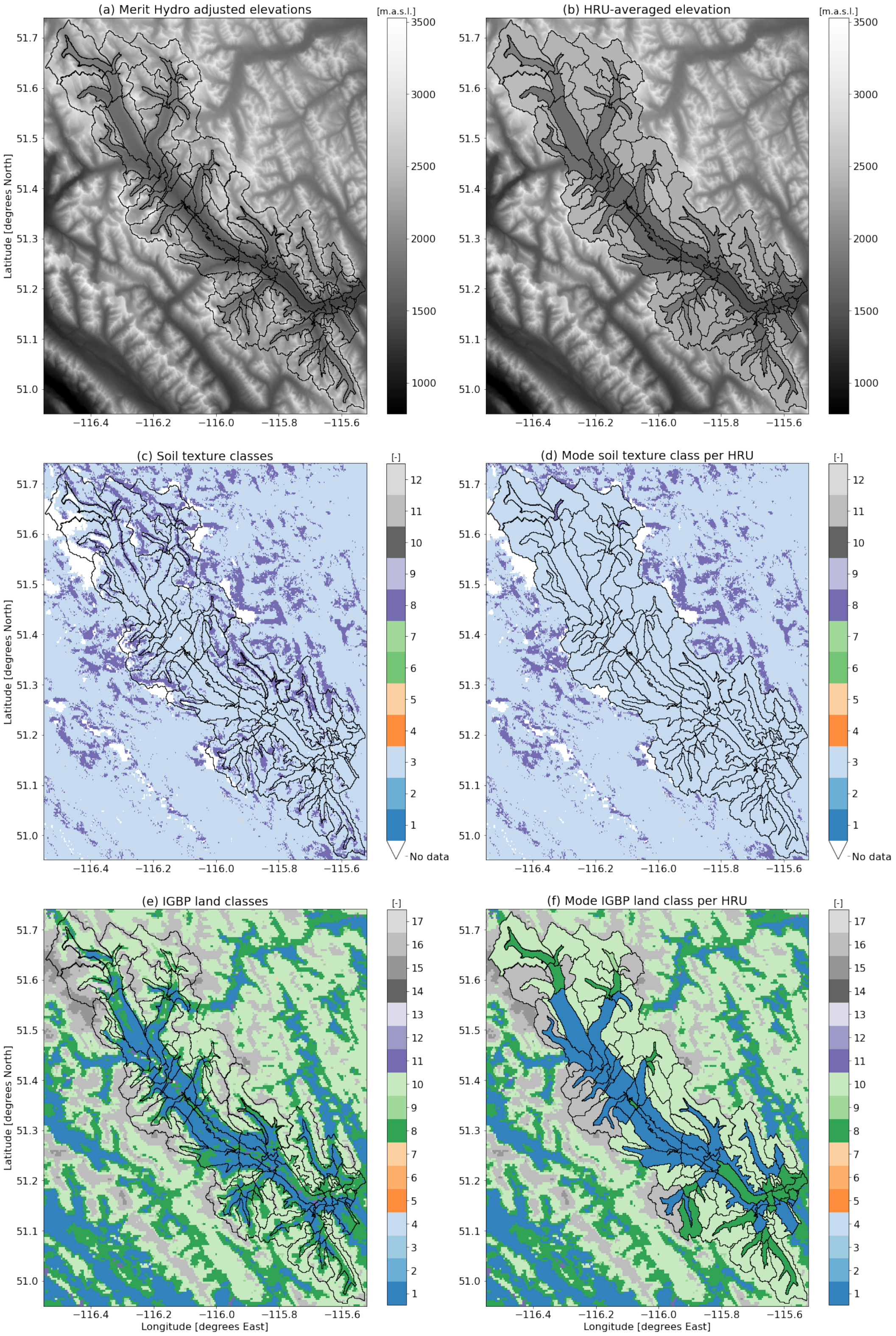


Figure A5. Mapping of geospatial parameter fields onto model elements. (a, b) MERIT Hydro adjusted elevations DEM source data and the mean elevation per HRU. (c, d) Soil texture classes derived from SOILGRIDS sand, silt and clay percentages and the most common class per HRU. (e, f) IGBP land classes from MODIS data and the most common class per HRU.

A25 Post-processing

Post-processing of model results in this example is limited to the code needed to generate the modeling domain figure in this manuscript (*./CWARHM/7_visualization*). Further visualization code may be added over time, as such code is created for specific experiments.

Appendix B Note on data accuracy

Our example workflow uses ERA5 forcing data, Merit Hydro DEM, SOILGRIDS-derived soil texture classes, and MODIS IGBP land classes for their global coverage. This enables global applications of the workflow. Such global datasets are based on a combination of observations and geospatial data processing methods to estimate data values for locations where no observations are available. These approaches may need to sacrifice local information content for global coverage and are not always able to utilize the most accurate local data available.

ERA5 is a reanalysis product from a data assimilating numerical weather prediction model. ERA5 precipitation estimates compare favorably to other global products at a daily resolution (Beck et al., 2019) but are typically not as accurate as local gauge or radar-based observations, especially in regions with complex topography (e.g., Amjad et al., 2020; Jiang et al., 2021; Tang et al., 2020; Xu et al., 2019). Jiang et al. (2020) show a similar reduced accuracy of ERA5 compared to station observations for direct and diffuse solar radiation estimates. Less is known about the accuracy of the remaining ERA5 forcing variables used in our workflow, and it is possible that the relatively coarse resolution of ERA5 data means that these variables may not be as accurate as local products.

The MERIT Hydro hydrologically adjusted elevation dataset (Yamazaki et al., 2019) is based on the MERIT DEM (Yamazaki et al., 2017), which itself is the result of applying an error-removal algorithm to existing space-borne DEMs. It is available globally at approximately 90 m spatial resolution. The MERIT Hydro data represent an advance over earlier products such as HydroSheds (Lehner et al., 2008), especially at higher latitudes, but some uncertainty in the produced hydrography data remains in regions with low topographic variation, with endorheic basins, with seasonally varying connectivity, and with channel bifurcations. The MERIT Hydro hydrologically adjusted elevations are a modification of the MERIT DEM that satisfies the condition "downstream pixels are not higher than upstream pixels". This procedure relies on a combination of correctly identifying endorheic basins, connections between sub-basins, and adjusting pixel elevations to create continuous flow paths. It is unknown to what extent this procedure affects the mean catchment elevation we derive from the hydrologically adjusted elevation. It is plausible that mean catchment elevations derived from this data will be less accurate in regions with rapidly varying topography, where catchment slopes are steep compared to the MERIT Hydro resolution.

The SoilGrids database uses observations of approximately 150,000 soil profiles, pseudo-observations that encode expert knowledge in a similar way to actual observed soil profiles, and machine learning to provide global estimates of various soil properties at a 250 m resolution. Ten-fold cross-validation of the resulting sand, silt, and clay percentage data used in our workflow shows that this approach explains approximately 75% of the variation in these soil properties. There is no systematic over or under prediction of these properties, but large differences between estimates and observations exist nonetheless in certain cases (Hengl et al., 2017).

MODIS MCD12Q1_v6 data uses a combination of random forests, bias and error correction based on ancillary data, and a hidden Markov Model approach to convert pre-processed satellite reflectance imagery into land cover classification categories. Ten-fold cross-validation of the resulting classification indicates that the IGBP classes used in our workflow are accurate in approximately two-thirds of cases. Misclassifications tend to occur in regions that contain substantial land cover variability at scales smaller than the 500 m MODIS resolution is provided at and along climatic gradients where the cover type changes gradually (Sulla-Menashe et al., 2019).

We therefore recommend that users replace our chosen global data products with more appropriate local data if such data are available and the project scope lies within the data domain. Due to the modular nature of the workflow, this replacement requires only minimal changes to the model configuration code. In terms of Figure 9, incorporating a different data set would require a new data-specific pre-processing module for which our existing workflow can serve as a guide. We emphasize that this workflow is intended to provide a baseline configuration upon which a user can improve. Our workflow does not contain any elements that compare the resulting simulations to observations to ascertain the quality of these simulations. A model setup generated through this workflow should thus not be assumed to be fit for a given purpose, unless shown to be so by the user's own model evaluation procedures.

Open Research

The latest version of the workflow code presented in this study is available on `https://github.com/CH-Earth/CWARHM` with the specific version used to generate Figures 7, 8, 10-14 via DOI, accessible under GNU GPL v3.0. **Note to reviewers: we intend to mint a DOI once the peer review process is complete**

The SUMMA (Clark et al., 2015a, 2015b, 2021c) version used for simulations in this paper can be identified by Git commit ID 3d17543 (North America domain and Bow at Banff domain). Source code accessible on: https://github.com/CH-Earth/summa

The mizuRoute (Mizukami et al., 2021, 2016) version used for simulations in this paper can be identified by Git commit ID c2de53d (North America domain) and Git commit ID d43066b (Bow at Banff domain). Source code accessible on: `https://github.com/ESCOMP/mizuRoute`

The single level ERA5 data (Hersbach et al., 2018) used as meteorological model input data are available at the Copernicus Climate Change Service (C3S) Climate Data Store (CDS) via `https://dx.doi.org/10.24381/cds.adbb2d47` under the *Licence to use Copernicus Products* (`https://cds.climate.copernicus.eu/api/v2/terms/static/licence-to-use-copernicus-products.pdf`; last access 2021-11-04).

The pressure level ERA5 data (Hersbach et al., 2017) used as meteorological model input data are available at the Copernicus Climate Change Service (C3S) Climate Data Store (CDS) via MARS request (no DOI) under *Licence to use Copernicus Products* (`https://cds.climate.copernicus.eu/api/v2/terms/static/licence-to-use-copernicus-products.pdf`; last access 2021-11-04). Data downloaded on 2021-04-17 for Bow at Banff test case; between 2020-11-14 and 2020-12-23 for North America test case.

The MERIT Hydro Hydrologically Adjusted Elevations (Yamazaki et al., 2019) used as Digital Elevation Model to determine mean catchment elevations is available at `http://hydro.iis.u-tokyo.ac.jp/~yamadai/MERIT_Hydro/` (last webpage access on 2021-11-04) as version v1.0.1 (no DOI available; data downloaded on 2021-04-17 for Bow at Banff test case, 2021-05-15 for North America test case), accessible under CC-BY-NC 4.0 or ODbL 1.0.

The MODIS MCD12Q1 V6 data (Friedl and Sulla-Menashe, 2019; Sulla-Menashe and Friedl, 2018) used to find a representative IGBP land cover class for each model element is available at the NASA EOSDIS Land Processes DAAC via `https://dx.doi.org/10.5067/MODIS/MCD12Q1.006`, with no restrictions on reuse, sale or redistribution.

The Global USDA-NRCS soil texture class map (Knoben, 2021) derived from the Soilgrids250m data set (Hengl et al., 2017) and used to find a representative USGS soil type class for each model element is available as a Hydroshare resource via `https://dx.doi.org/10.4211/hs.1361509511e44adfba814f6950c6e742`, under ODbL v1.0.

The shapefile that contains the catchment delineation for the Bow At Banff test case is derived from the MERIT Hydro basins data set (Lin et al., 2019) and available as part of the workflow repository. The shapefile that contains the catchment delineation for the North America test case is derived from the MERIT Hydro basins data set, and ... **Note to reviewers: we are in the processing of ensuring that this shapefile will be publicly available.**

Acknowledgements

We are thankful for the contributions of Guoqiang Tang, who confirmed that our code is functional on macOS and the general comments for workflow improvement given by Guoqiang Tang, Dave Casson, Hongli Liu, Jim Freer and Hannah Burdett. We are also thankful for the efforts of Reza Zolfaghari and Kyle Klenk who assisted in understanding and solving specific numerical issues, and to Louise Arnal for feedback on the workflow schematics. Wouter Knoben, Martyn Clark, Shervan Gharari, Chris Marsh and Ray Spiteri were supported by the Global Water Futures program, University of Saskatchewan. The work of David Tarboton was supported by the US National Science Foundation under collaborative grants ICER 1928369 and OAC 1835569. The work of Jerad Bales was supported by NSF grant EAR 1849458. NCAR co-author contributions were supported by the US Army Corps of Engineers Responses to Climate Change program and the US Bureau of Reclamation under Cooperative Agreement R16AC00039.

References

Amjad, M., Yilmaz, M.T., Yucel, I., Yilmaz, K.K., 2020. Performance evaluation of satellite- and model-based precipitation products over varying climate and complex topography. J. Hydrol. 584, 124707. https://doi.org/10.1016/j.jhydrol.2020.124707

Añel, J.A., 2017. Comment on "Most computational hydrology is not reproducible, so is it really science?" by Christopher Hutton et al.: NECESSARY CONDITIONS FOR REPRODUCIBILITY. Water Resour. Res. 53, 2572–2574. https://doi.org/10.1002/2016WR020190

Archfield, S.A., Clark, M., Arheimer, B., Hay, L.E., McMillan, H., Kiang, J.E., Seibert, J., Hakala, K., Bock, A., Wagener, T., Farmer, W.H., Andréassian, V., Attinger, S., Viglione, A., Knight, R., Markstrom, S., Over, T., 2015. Accelerating advances in continental domain hydrologic modeling. Water Resour. Res. 51, 10078–10091. https://doi.org/10.1002/2015WR017498

Arsenault, R., Poulin, A., Côté, P., Brissette, F., 2014. Comparison of Stochastic Optimization Algorithms in Hydrological Model Calibration. J. Hydrol. Eng. 19, 1374–1384. https://doi.org/10.1061/(ASCE)HE.1943-5584.0000938

Bandaragoda, C., Castronova, A., Istanbulluoglu, E., Strauch, R., Nudurupati, S.S., Phuong, J., Adams, J.M., Gasparini, N.M., Barnhart, K., Hutton, E.W.H., Hobley, D.E.J., Lyons, N.J., Tucker, G.E., Tarboton, D.G., Idaszak, R., Wang, S., 2019. Enabling Collaborative Numerical Modeling in Earth Sciences using Knowledge Infrastructure. Environ. Model. Softw. 120, 104424. https://doi.org/10.1016/j.envsoft.2019.03.020

Bast, R., 2019. A FAIRer future. Nat. Phys. 15, 728–730. https://doi.org/10.1038/s41567-019-0624-3

Bavay, M., Reisecker, M., Egger, T., Korhammer, D., 2020. Inishell 2.0: Semantically driven automatic GUI generation forscientific models (preprint). Hydrology. https://doi.org/10.5194/gmd-2020-339

Beck, H.E., Pan, M., Roy, T., Weedon, G.P., Pappenberger, F., van Dijk, A.I.J.M., Huffman, G.J., Adler, R.F., Wood, E.F., 2019. Daily evaluation of 26 precipitation datasets using Stage-IV gauge-radar data for the CONUS. Hydrol. Earth Syst. Sci. 23, 207–224. https://doi.org/10.5194/hess-23-207-2019

Benham, E., Ahrens, R.J., Nettleton, W.D., 2009. Clarification of Soil Texture Class Boundaries.

Bennett, A., Hamman, J., Nijssen, B., 2020. MetSim: A Python package for estimation and disaggregation of meteorological data. J. Open Source Softw. 5, 2042. https://doi.org/10.21105/joss.02042

Blair, G.S., Beven, K., Lamb, R., Bassett, R., Cauwenberghs, K., Hankin, B., Dean, G., Hunter, N., Edwards, L., Nundloll, V., Samreen, F., Simm, W., Towe, R., 2019. Models of everywhere revisited: A technological perspective. Environ. Model. Softw. 122, 104521. https://doi.org/10.1016/j.envsoft.2019.104521

Blodgett, D., Dornblut, I., 2018. OGC® WaterML 2: Part 3 - Surface Hydrology Features (HY_Features) - Conceptual Model [WWW Document]. URL http://docs.opengeospatial.org/is/14-111r6/14-111r6.html#_catchment_area (accessed 12.2.21).

Blöschl, G., Bárdossy, A., Koutsoyiannis, D., Kundzewicz, Z.W., Littlewood, I., Montanari, A., Savenije, H., 2014. On the future of journal publications in hydrology. Water Resour. Res. 50, 2795–2797. https://doi.org/10.1002/2014WR015613

Bock, a. R., Hay, L.E., McCabe, G.J., Markstrom, S.L., Atkinson, R.D., 2015. Parameter regionalization of a monthly water balance model for the conterminous United States. Hydrol. Earth Syst. Sci. Discuss. 12, 10023–10066. https://doi.org/10.5194/hessd-12-10023-2015

Chaney, N., Fisher, C., 2021. chaneyn/geospatialtools: geospatialtools_nov2021. Zenodo. https://doi.org/10.5281/ZENODO.5676031

Choi, Y.-D., Goodall, J.L., 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, 104888. https://doi.org/10.1016/j.envsoft.2020.10

Chuah, J., Deeds, M., Malik, T., Choi, Y., Goodall, J.L., 2020. Documenting Computing Environments for Reproducible Experiments, in: Parallel Computing: Technology Trends, Advances in Parallel Computing. IOS Press, Amsterdam, Netherlands, pp. 756–765.

Clark, M.P., Luce, C.H., AghaKouchak, A., Berghuijs, W., David, C.H., Duan, Q., Ge, S., van Meerveld, I., Zheng, C., Parlange, M.B., Tyler, S.W., 2021a. Open Science: Open Data, Open Models, …and Open Publications? Water Resour. Res. 57. https://doi.org/10.1029/2020WR029480

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., 2015a. 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., Nijssen, B., Lundquist, J.D., Kavetski, D., Rupp, D.E., Woods, R.A., Freer, J.E., Gutmann, E.D., Wood, A.W., Gochis, D.J., Rasmussen, R.M., Tarboton, D.G., Mahat, V., Flerchinger, G.N., Marks, D.G., 2015b. A unified approach for process-based hydrologic modeling: 2. Model implementation and case studies. Water Resour. Res. 51, 2515–2542. https://doi.org/10.1002/2015WR017200

Clark, M.P., Vogel, R.M., Lamontagne, J.R., Mizukami, N., Knoben, W.J.M., Tang, G., Gharari, S., Freer, J.E., Whitfield, P.H., Shook, K.R., Papalexiou, S.M., 2021b. The Abuse of Popular Performance Metrics in Hydrologic Modeling. Water Resour. Res. 57. https://doi.org/10.1029/2020W

Clark, M.P., Zolfaghari, R., Green, K.R., Trim, S., Knoben, W.J.M., Bennett, A., Nijssen, B., Ireson, A., Spiteri, R.J., 2021c. The numerical implementation of land models: Problem formulation and laugh tests. J. Hydrometeorol. https://doi.org/10.1175/JHM-D-20-0175.1

Copernicus Climate Change Service (C3S), 2017. ERA5: Fifth generation of ECMWF atmospheric reanalyses of the global climate.

Cullen, R.M., Marshall, S.J., 2011. Mesoscale Temperature Patterns in the Rocky Mountains and Foothills Region of Southern Alberta. Atmosphere-Ocean 49, 189–205. https://doi.org/10.1080/0705590

Dutra, E., Muñoz-Sabater, J., Boussetta, S., Komori, T., Hirahara, S., Balsamo, G., 2020. Environmental Lapse Rate for High-Resolution Land Surface Downscaling: An Application to ERA5. Earth Space Sci. 7. https://doi.org/10.1029/2019EA000984

Essawy, B.T., Goodall, J.L., 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, 104753. 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 Space Sci. 3, 163–175. https://doi.org/10.1002/2015EA000139

Famiglietti, J.S., Murdoch, L., Lakshmi, V., Arrigo, J., Hooper, R., 2011. Establishing a Framework for Community Modeling in Hydrologic Science, Report from the 3rd Workshop on a Community Hydrologic Modeling Platform (CHyMP): A Strategic and Implementation Plan. Beckman Center of the National Academies, University of California at Irvine.

Farr, T.G., Rosen, P.A., Caro, E., Crippen, R., Duren, R., Hensley, S., Kobrick, M., Paller, M., Rodriguez, E., Roth, L., Seal, D., Shaffer, S., Shimada, J., Umland, J., Werner, M., Oskin, M., Burbank, D., Alsdorf, D., 2007. The Shuttle Radar Topography Mission. Rev. Geophys. 45, RG2004. https://doi.org/10.1029/2005RG000183

Friedl, M., Sulla-Menashe, D., 2019. MCD12Q1 MODIS/Terra+Aqua Land Cover Type Yearly L3 Global 500m SIN Grid V006. https://doi.org/10.5067/MODIS/MCD12Q1.006

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, 104731. https://doi.org/10.1016/j.envsoft.2020.104731

Gharari, S., Clark, M.P., Mizukami, N., Knoben, W.J.M., Wong, J.S., Pietroniro, A., 2020. Flexible vector-based spatial configurations in land models. Hydrol. Earth Syst. Sci. 24, 5953–5971. https://doi.org/10.5194/hess-24-5953-2020

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/10.1016/j.envsoft.2020.104623

Gil, Y., David, C.H., Demir, I., Essawy, B.T., Fulweiler, R.W., Goodall, J.L., Karlstrom, L., Lee, H., Mills, H.J., Oh, J., Pierce, S.A., Pope, A., Tzeng, M.W., Villamizar, S.R., Yu, X., 2016. Toward the Geoscience Paper of the Future: Best practices for documenting and sharing research from data to software to provenance. Earth Space Sci. 3, 388–415. https://doi.org/10.1002/2015EA000136

Gupta, H.V., Wagener, T., Liu, Y., 2008. Reconciling theory with observations : elements of a diagnostic approach to model evaluation. Hydrol. Process. 3813, 3802–3813. https://doi.org/10.1002/hyp.698

Hamman, J.J., Nijssen, B., Bohn, T.J., Gergel, D.R., Mao, Y., 2018. The Variable Infiltration Capacity model version 5 (VIC-5): infrastructure improvements for new applications and reproducibility. Geosci. Model Dev. 11, 3481–3496. https://doi.org/10.5194/gmd-11-3481-2018

Havens, S., Marks, D., Sandusky, M., Hedrick, A., Johnson, M., Robertson, M., Trujillo, E., 2020. Automated Water Supply Model (AWSM): Streamlining and standardizing application of a physically based snow model for water resources and reproducible science. Comput. Geosci. 144, 104571. https://doi.org/10.1016/j.cageo.2020.104571

Hengl, T., Mendes de Jesus, J., Heuvelink, G.B.M., Ruiperez Gonzalez, M., Kilibarda, M., Blagotić, A., Shangguan, W., Wright, M.N., Geng, X., Bauer-Marschallinger, B., Guevara, M.A., Vargas, R., MacMillan, R.A., Batjes, N.H., Leenaars, J.G.B., Ribeiro, E., Wheeler, I., Mantel, S., Kempen, B., 2017. SoilGrids250m: Global gridded soil information based on machine learning. PLOS ONE 12, e0169748. https://doi.org/10.1371/journal.pone.0169748

Hersbach, H., Bell, B., Berrisford, P., Biavati, G., Horányi, A., Muñoz Sabater, J., Nicolas, J., Peubey, C., Radu, R., Rozum, I., Schepers, D., Simmons, A., Soci, C., Dee, D., Thépaut, J.-N., 2018. ERA5 hourly data on single levels from 1979 to present. https://doi.org/10.24381/cds.adbb

Hersbach, H., Bell, B., Berrisford, P., Hirahara, S., Horányi, A., Muñoz-Sabater, J., Nicolas, J., Peubey, C., Radu, R., Schepers, D., Simmons, A., Soci, C., Abdalla, S., Abellan, X., Balsamo, G., Bechtold, P., Biavati, G., Bidlot, J., Bonavita, M., Chiara, G., Dahlgren, P., Dee, D., Diamantakis, M., Dragani, R., Flemming, J., Forbes, R., Fuentes, M., Geer, A., Haimberger, L., Healy, S., Hogan, R.J., Hólm, E., Janisková, M., Keeley, S., Laloyaux, P., Lopez, P., Lupu, C., Radnoti, G., Rosnay, P., Rozum, I., Vamborg, F., Villaume, S., Thépaut, J., 2020. The ERA5 global reanalysis. Q. J. R. Meteorol. Soc. 146, 1999–2049. https://doi.org/10.1002/qj.3803

Hersbach, H., Bell, B., Berrisford, P., Hirahara, S., Horányi, A., Muñoz-Sabater, J., Nicolas, J., Peubey, C., Radu, R., Schepers, D., Simmons, A., Soci, C., Abdalla, S., Abellan, X., Balsamo, G., Bechtold, P., Biavati, G., Bidlot, J., Bonavita, M., De Chiara, G., Dahlgren, P., Dee, P., Diamantakis, M., Dragani, R., Flemming, J., Forbes, R., Fuentes, M., Geer, A., Haimberger, L., Healy, S., Hogan, R.J., Hólm, E., Janisková, M., Keeley, S., Laloyaux, P., Lopez, P., Lupu, C., Radnoti, G., de Rosnay, P., Rozum, I., Vamborg, F., Villaume, S., Thépaut, J.-N., 2017. Complete ERA5: Fifth generation of ECMWF atmospheric reanalyses of the global climate.

Husain, S.Z., Alavi, N., Bélair, S., Carrera, M., Zhang, S., Fortin, V., Abrahamowicz, M., Gauthier, N., 2016. The Multibudget Soil, Vegetation, and Snow (SVS) Scheme for Land Surface Parameterization: Offline Warm Season Evaluation. J. Hydrometeorol. 17, 2293–2313. https://doi.org/10.1175/JHM-D-15-0228.1

Hut, R.W., van de Giesen, N.C., Drost, N., 2017. Comment on “Most computational hydrology is not reproducible, so is it really science?” by Christopher Hutton et al.: Let hydrologists learn the latest computer science by working with Research Software Engineers (RSEs) and not reinvent the waterwheel ourselves. Water Resour. Res. 53, 4524–4526. https://doi.org/10.1002/2017WR020665

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

Jiang, H., Yang, Y., Bai, Y., Wang, H., 2020. Evaluation of the Total, Direct, and Diffuse Solar Radiations From the ERA5 Reanalysis Data in China. IEEE Geosci. Remote Sens. Lett. 17, 47–51. https://doi.org/10.1109/LGRS.2019.2916410

Jiang, Q., Li, W., Fan, Z., He, X., Sun, W., Chen, S., Wen, J., Gao, J., Wang, J., 2021. Evaluation of the ERA5 reanalysis precipitation dataset over Chinese Mainland. J. Hydrol. 595, 125660. https://doi.org/10.1016/j.jhydrol.2020.125660

Knoben, W.J.M., 2021. Global USDA-NRCS soil texture class map. https://doi.org/10.4211/hs.1361509511e44adf

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

Lehner, B., Grill, G., 2013. Global river hydrography and network routing: baseline data and new approaches to study the world's large river systems: GLOBAL RIVER HYDROGRAPHY AND NETWORK ROUTING. Hydrol. Process. 27, 2171–2186. https://doi.org/10.1002/hyp.9740

Lehner, B., Verdin, K., Jarvis, A., 2008. New Global Hydrography Derived From Spaceborne Elevation Data. Eos Trans. Am. Geophys. Union 89, 93. https://doi.org/10.1029/2008EO100001

Leonard, L., Duffy, C., 2016. Visualization workflows for level-12 HUC scales: Towards an expert system for watershed analysis in a distributed computing environment. Environ. Model. Softw. 78, 163–178. https://doi.org/10.1016/j.envsoft.2016.01.001

Leonard, L., Duffy, C.J., 2014. Automating data-model workflows at a level 12 HUC scale: Watershed modeling in a distributed computing environment. Environ. Model. Softw. 61, 174–190. https://doi.org/10.1016/j.envsoft.2014.07.015

Leonard, L., Duffy, C.J., 2013. Essential Terrestrial Variable data workflows for distributed water resources modeling. Environ. Model. Softw. 50, 85–96. https://doi.org/10.1016/j.envsoft.2013.09.003

Liang, X., Lettenmaier, D.P., Wood, E.F., Burges, S.J., 1994. A simple hydrologically based model of land surface water and energy fluxes for general circulation models. J. Geophys. Res. 99, 14415–14428.

Lin, P., Pan, M., Beck, H.E., Yang, Y., Yamazaki, D., Frasson, R., David, C.H., Durand, M., Pavelsky, T.M., Allen, G.H., Gleason, C.J., Wood, E.F., 2019. Global Reconstruction of Naturalized River Flows at 2.94 Million Reaches. Water Resour. Res. 55, 6499–6516. https://doi.org/10.1029/2019WR025287

McMillan, H.K., 2021. A review of hydrologic signatures and their applications. WIREs Water 8. https://doi.org/10.1002/wat2.1499

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.: REPRODUCING COMPUTATIONAL STUDIES. Water Resour. Res. 53, 2568–2569. https://doi.org/10.1002/2016WR020208

Messager, M.L., Lehner, B., Grill, G., Nedeva, I., Schmitt, O., 2016. Estimating the volume and age of water stored in global lakes using a geo-statistical approach. Nat. Commun. 7, 13603. https://doi.org/10.1038/ncomms13603

Miles, B., Band, L.E., 2015. Ecohydrology Models without Borders?: Using Geospatial Web Services in EcohydroLib Workflows in the United States and Australia, in: Denzer, R., Argent, R.M., Schimak, G., Hřebíček, J. (Eds.), Environmental Software Systems. Infrastructures, Services and Applications, IFIP Advances in Information and Communication Technology. Springer International Publishing, Cham, pp. 311–320. https://doi.org/10.1007/978-3-319-15994-2_31

Miles, B.C., 2014. SMALL-SCALE RESIDENTIAL STORMWATER MANAGEMENT IN URBANIZED WATERSHEDS: A GEOINFORMATICS-DRIVEN ECOHYDROLOGY MODELING APPROACH. A dissertation submitted to the faculty at the University of North Carolina at Chapel Hill, Chapel Hill 2014.

Minder, J.R., Mote, P.W., Lundquist, J.D., 2010. Surface temperature lapse rates over complex terrain: Lessons from the Cascade Mountains. J. Geophys. Res. 115, D14122. https://doi.org/10.1029/2009JD013493

Mizukami, N., Clark, M.P., Gharari, S., Kluzek, E., Pan, M., Lin, P., Beck, H.E., Yamazaki, D., 2021. A Vector-Based River Routing Model for Earth System Models: Parallelization and Global Applications. J. Adv. Model. Earth Syst. 13. https://doi.org/10.1029/2020MS002434

Mizukami, N., Clark, M.P., Sampson, K., Nijssen, B., Mao, Y., McMillan, H., Viger, R.J., Markstrom, S.L., Hay, L.E., Woods, R., Arnold, J.R., Brekke, L.D., 2016. mizuRoute version 1: a river network routing tool for a continental domain water resources applications. Geosci. Model Dev. 9, 2223–2238. https://doi.org/10.5194/gmd-9-2223-2016

Mizukami, N., Rakovec, O., Newman, A.J., Clark, M.P., Wood, A.W., Gupta, H.V., Kumar, R., 2019. On the choice of calibration metrics for “high-flow” estimation using hydrologic models. Hydrol. Earth Syst. Sci. 23, 2601–2614. https://doi.org/10.5194/hess-23-2601-2019

Murphy, A.H., 1988. Skill Scores Based on the Mean Square Error and Their Relationships to the Correlation Coefficient. Mon. WEATHER Rev. 116, 2417–2424. https://doi.org/10.1175/1520-0493(1988)116%3C2417:SSBOTM%3E2.0.CO;2

Nash, J.E., Sutcliffe, J.V., 1970. River flow forecasting through conceptual models part I — A discussion of principles. J. Hydrol. 10, 282–290. https://doi.org/10.1016/0022-1694(70)90255-6

NCAR Research Applications Laboratory, RAL, 2021. Noah-Multiparameterization Land Surface Model (Noah-MP LSM) [WWW Document]. URL https://ral.ucar.edu/solutions/products/noah-multiparameterization-land-surface-model-noah-mp-lsm (accessed 11.15.21).

Olden, J.D., Poff, N.L., 2003. Redundancy and the choice of hydrologic indices for characterizing streamflow regimes. River Res. Appl. 19, 101–121. https://doi.org/10.1002/rra.700

Pechlivanidis, I.G., Jackson, B.M., McIntyre, N.R., Wheater, H.S., 2011. Catchment scale hydrological modelling: a review of model types, calibration approaches and uncertainty analysis methods in the context of recent developments in technology and applications. Glob. NEST 13, 193–214.

Pietroniro, A., Fortin, V., Kouwen, N., Neal, C., Turcotte, R., Davison, B., Verseghy, D., Soulis, E.D., Caldwell, R., Evora, N., Pellerin, P., 2007. Development of the MESH modelling system for hydrological ensemble forecasting of the Laurentian Great Lakes at the regional scale. Hydrol. Earth Syst. Sci. 11, 1279–1294. https://doi.org/10.5194/hess-11-1279-2007

Pushpalatha, R., Perrin, C., Moine, N.L., Andréassian, V., 2012. A review of efficiency criteria suitable for evaluating low-flow simulations. J. Hydrol. 420–421, 171–182. https://doi.org/10.1016/j.jhydr

QGIS Development Team, 2021. QGIS Geographic Information System. QGIS Association.

Samaniego, L., Kumar, R., Attinger, S., 2010. Multiscale parameter regionalization of a grid-based hydrologic model at the mesoscale. Water Resour. Res. 46, 1–25. https://doi.org/10.1029/2008WR00

Sandve, G.K., Nekrutenko, A., Taylor, J., Hovig, E., 2013. Ten Simple Rules for Reproducible Computational Research. PLoS Comput. Biol. 9, e1003285. https://doi.org/10.1371/journal.pcbi.1003285

Sazib, N.S., 2016. Physically Based Modeling of the Impacts of Climate Change on Streamflow Regime. https://doi.org/10.26076/7BBE-E62D

Sen Gupta, A., Tarboton, D.G., Hummel, P., Brown, M.E., Habib, S., 2015. Integration of an energy balance snowmelt model into an open source modeling framework. Environ. Model. Softw. 68, 205–218. https://doi.org/10.1016/j.envsoft.2015.02.017

Slater, L.J., Thirel, G., Harrigan, S., Delaigue, O., Hurley, A., Khouakhi, A., Prosdocimi, I., Vitolo, C., Smith, K., 2019. Using R in hydrology: a review of recent developments and future directions. Hydrol. Earth Syst. Sci. 23, 2939–2963. https://doi.org/10.5194/hess-23-2939-2019

Stagge, J.H., Rosenberg, D.E., Abdallah, A.M., Akbar, H., Attallah, N.A., James, R., 2019. Assessing data availability and research reproducibility in hydrology and water resources. Sci. Data 6, 190030. https://doi.org/10.1038/sdata.2019.30

Stodden, V., Miguez, S., 2013. Best Practices for Computational Science: Software Infrastructure and Environments for Reproducible and Extensible Research. SSRN Electron. J. https://doi.org/10.2139/ssrn.2322276

Sulla-Menashe, D., Friedl, M.A., 2018. User Guide to Collection 6 MODIS Land Cover (MCD12Q1 andMCD12C1) Product (User Guide).

Sulla-Menashe, D., Gray, J.M., Abercrombie, S.P., Friedl, M.A., 2019. Hierarchical mapping of annual global land cover 2001 to present: The MODIS Collection 6 Land Cover product. Remote Sens. Environ. 222, 183–194. https://doi.org/10.1016/j.rse.2018.12.013

Tadono, T., Nagai, H., Ishida, H., Oda, F., Naito, S., Minakawa, K., Iwamoto, H., 2016. GENERATION OF THE 30 M-MESH GLOBAL DIGITAL SURFACE MODEL BY ALOS PRISM. ISPRS - Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. XLI-B4, 157–162. https://doi.org/10.5194/isprsarchives-XLI-B4-157-2016

Tang, G., Clark, M.P., Papalexiou, S.M., Ma, Z., Hong, Y., 2020. Have satellite precipitation products improved over last two decades? A comprehensive comparison of GPM IMERG with nine satellite and reanalysis datasets. Remote Sens. Environ. 240, 111697. https://doi.org/10.1016/j.rse.2020.111697

Tarboton, D.G., Horsburgh, J.S., Maidment, D.R., Whiteaker, T., Zaslavsky, I., Piasecki, M., Goodall, J., Valentine, D., Whitenack, T., 2009. Development of a community hydrologic information system. Presented at the 18th World IMACS Congress and MODSIM09 International Congress on Modelling and Simulation, Modelling and Simulation Society of Australia and New Zealand and International Association for Mathematics and Computers in Simulation, pp. 988–994.

Tesfa, T.K., Tarboton, D.G., Watson, D.W., Schreuders, K.A.T., Baker, M.E., Wallace, R.M., 2011. Extraction of hydrological proximity measures from DEMs using parallel processing. Environ. Model. Softw. 26, 1696–1709. https://doi.org/10.1016/j.envsoft.2011.07.018

Tiesinga, E., Mohr, P.J., Newell, D.B., Taylor, B.N., 2019. The 2018 CODATA Recommended Values of the Fundamental Physical Constants. National Institute of Standards and Technology, Gaithersburg, MD 20899.

U.S. Department of Agriculture: Agricultural Research Service (USDA ARS), 2021. ROSETTA Class Average Hydraulic Parameters [WWW Document]. URL https://www.ars.usda.gov/pacific-west-area/riverside-ca/agricultural-water-efficiency-and-salinity-research-unit/docs/model/rosetta-class-average-hydraulic-parameters/ (accessed 11.15.21).

Vanderborght, J., Kasteel, R., Herbst, M., Javaux, M., Thiéry, D., Vanclooster, M., Mouvet, C., Vereecken, H., 2005. A Set of Analytical Benchmarks to Test Numerical Models of Flow and Transport in Soils. Vadose Zone J. 4, 206. https://doi.org/10.2136/vzj2005.0206

Vorobevskii, I., Kronenberg, R., Bernhofer, C., 2020. Global BROOK90 R Package: An Automatic Framework to Simulate the Water Balance at Any Location. Water 12, 2037. https://doi.org/10.3390/w12072037

Wallace, J.M., Hobbs, P.V., 2006. Atmospheric science: an introductory survey, 2nd ed. ed, International geophysics series. Elsevier Academic Press, Amsterdam ; Boston.

Weiler, M., Beven, K., 2015. Do we need a community hydrological model? Water Resour. Res. 51. https://doi.org/10.1002/2014WR016731

Wilkinson, M.D., Dumontier, M., Aalbersberg, Ij.J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L.B., Bourne, P.E., Bouwman, J., Brookes, A.J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C.T., Finkers, R., Gonzalez-Beltran, A., Gray, A.J.G., Groth, P., Goble, C., Grethe, J.S., Heringa, J., 't Hoen, P.A.C., Hooft, R., Kuhn, T., Kok, R., Kok, J., Lusher, S.J., Martone, M.E., Mons, A., Packer, A.L., Persson, B., Rocca-Serra, P., Roos, M., van Schaik, R., Sansone, S.-A., Schultes, E., Sengstag, T., Slater, T., Strawn, G., Swertz, M.A., Thompson, M., van der Lei, J., van Mulligen, E., Velterop, J., Waagmeester, A., Wittenburg, P., Wolstencroft, K., Zhao, J., Mons, B., 2016. The FAIR Guiding Principles for scientific data management and stewardship. Sci. Data 3, 160018. https://doi.org/10.1038/sdata.2016.18

Xu, X., Frey, S.K., Boluwade, A., Erler, A.R., Khader, O., Lapen, D.R., Sudicky, E., 2019. Evaluation of variability among different precipitation products in the Northern Great Plains. J. Hydrol. Reg. Stud. 24, 100608. https://doi.org/10.1016/j.ejrh.2019.100608

Yamazaki, D., Ikeshima, D., Sosa, J., Bates, P.D., Allen, G.H., Pavelsky, T.M., 2019. MERIT Hydro: A High-Resolution Global Hydrography Map Based on Latest Topography Dataset. Water Resour. Res. 55, 5053–5073. https://doi.org/10.1029/2019WR024873

Yamazaki, D., Ikeshima, D., Tawatari, R., Yamaguchi, T., O'Loughlin, F., Neal, J.C., Sampson, C.C., Kanae, S., Bates, P.D., 2017. A high-accuracy map of global terrain elevations: Accurate Global Terrain Elevation map. Geophys. Res. Lett. 44, 5844–5853. https://doi.org/10.1002/2017GL072874