AGU PUBLICATIONS

Earth and Space Science

RESEARCH ARTICLE

10.1002/2016ea000237

Special Section:
Geoscience papers of the Future

Key Points:

- This paper provides a simple example of a reproducible hydrologic modeling study
- All of the data, models and processing scripts used in this study are shared online
- Customized data preparation scripts are typically required due to data heterogeneity

Supporting Information:

- Supporting Information S1

Correspondence to:
S. D. Peckham,
scott.peckham@colorado.edu

Citation:
Peckham, S. D., M. Stoica, E. Jafarov, A. Endalamaw, and W. R. Bolton (2017), Reproducible, component-based modeling with TopoFlow, a spatial hydrologic modeling toolkit, *Earth and Space Science*, *4*, doi:10.1002/2016ea000237.

Received 2 NOV 2016
Accepted 18 APR 2017
Accepted article online 26 APR 2017

Reproducible, component-based modeling with TopoFlow, a spatial hydrologic modeling toolkit

Scott D. Peckham[1], **Maria Stoica**[1], **Elchin Jafarov**[2], **Abraham Endalamaw**[3], **and W. Robert Bolton**[3]

[1]Institute of Arctic and Alpine Research, University of Colorado Boulder, Boulder, Colorado, USA, [2]Los Alamos National Laboratory, Los Alamos, New Mexico, USA, [3]International Arctic Research Center, University of Alaska, Fairbanks, Alaska, USA

Abstract Modern geoscientists have online access to an abundance of different data sets and models, but these resources differ from each other in myriad ways and this heterogeneity works against interoperability as well as reproducibility. The purpose of this paper is to illustrate the main issues and some best practices for addressing the challenge of reproducible science in the context of a relatively simple hydrologic modeling study for a small Arctic watershed near Fairbanks, Alaska. This study requires several different types of input data in addition to several, coupled model components. All data sets, model components and processing scripts (e.g., for preparation of data and figures, and for analysis of model output) are fully documented and made available online at persistent URLs. Similarly, all source codes for the models and scripts are open source, version controlled, and made available online via GitHub. Each model component has a Basic Model Interface to simplify coupling and its own HTML help page that includes a list of all equations and variables used. The set of all model components (TopoFlow) has also been made available as a Python package for easy installation. Three different graphical user interfaces for setting up TopoFlow runs are described, including one that allows model components to run and be coupled as web services.

1. Introduction

Observational data and predictive models (or experimentation and theory) are the two pillars of science, although both are significantly impacted by computation [*Vardi*, 2010]. Physically based, mathematical models summarize our current best knowledge of how physical systems operate, and they are used to build computational models that predict future states from initial conditions. Data sets and computational models therefore represent two fundamental resource types that geoscientists use to work on problems of societal interest, such as watershed management and the effects of climate change. While modern geoscientists have online access to an abundance of different data sets and models, these resources differ from each other in myriad ways and this heterogeneity works against interoperability. Interoperability is typically necessary, however, because a single data set or model is seldom sufficient to tackle a nontrivial geoscience problem. For example, models typically require several different types of input data which they may obtain from reading input files or from other models. It is therefore often necessary to couple together many different heterogeneous resources into a computational workflow, and geoscientists spend a large fraction of their time just setting up and executing these workflows. An extensive, empirical analysis of scientific workflows [*Garijo et al.*, 2013] (and references therein) suggests that geoscientists typically spend between 60% and 80% of their time on dealing with such issues, leaving the remainder as the time available for the science. The phrase *data friction* [*Edwards*, 2010; *Edwards et al.*, 2011] has also been used to describe this problem.

This challenge of interoperability leads to significant complexity and is one of the main barriers to *reproducibility*, that is, the ability to reproduce a scientific study or experiment. Reproducibility is considered to be one of the cornerstones of science, but complex workflows often cannot be replicated unless they are fully described and documented. In recent years this topic has been receiving increased attention. *Hutton et al.* [2016] cite examples from a broad array of scientific disciplines and then focus on this issue in the context of computational hydrology. Since most modern scientific workflows rely on numerous *digital resources*, such as data sets, data preparation and analysis software, and computational models, it is now recognized that simply describing a workflow is not enough—all of the digital resources used in a study should also be made available with persistent identifiers (i.e., URLs and DOIs).

This paper begins with a description of a spatial, hydrologic model called TopoFlow developed by the first author and colleagues over the last 16 years. This is followed by an example application to a small, Arctic watershed in the Caribou-Poker Creek Research Watershed (CPCRW). This application starts with a description of the study site and then walks through all of the steps involved in obtaining the required input data sets, preparing them for use by TopoFlow, setting up a model run, executing the model, and then analyzing the results. By focusing on late summer rainfall events, the dominant hydrologic processes are rainfall and infiltration—in fact, a large percentage of the rainfall volume is lost to infiltration and does not contribute to the volume flow rate (discharge) at the basin outlet. A well-known, simplified, but physically based model for the infiltration process, known as the Green-Ampt method, is used within the model and also in our interpretation of the results.

2. Overview of the TopoFlow Model Toolkit

TopoFlow is a spatially distributed, process-based and open-source hydrologic model [*Peckham*, 2009a]. Development of TopoFlow started in 2000 through a collaboration between Peckham (University of Colorado, Boulder) and several colleagues at WERC, the Water and Environmental Research Center (University of Alaska, Fairbanks). Peckham had just developed a spatially distributed rainfall-runoff model that used the D8 method for determining flow directions over topography given as a digital elevation model (DEM). That model distinguished between DEM grid cells on hillslopes and those containing channels, using overland flow for the former and open-channel flow hydraulics for the latter [*Henderson*, 1966]. Each channel grid cell contained a prismatic channel with its own trapezoidal cross section and roughness parameters. It supported three different methods of channel flow routing, namely kinematic wave, diffusive wave, and dynamic wave. It also provided the option of using either Manning's formula or the logarithmic law of the wall to model flow resistance. Hinzman's group at WERC had just published a paper on their ARHYTHM model [*Zhang et al.*, 2000], a spatially distributed hydrologic model for use with Arctic watersheds with an advanced treatment of thermal processes. ARHYTHM used a computational grid of triangles and supported multiple treatments of snowmelt and evaporation, based on many years of prior work in the Arctic, e.g., *Hinzman et al.* [1996] and *Hinzman et al.* [1998]. ARHYTHM supported energy balance treatments of both snowmelt and evaporation in cases where shortwave and longwave radiation measurements were available. It also supported simplified treatments of these two processes that required less input data, namely the Degree-Day method for snowmelt and the Priestley-Taylor method for evaporation.

2.1. Single-Application, Interactive Data Language (IDL) Version of TopoFlow With GUI

This collaboration led to the merging of the two models into a single model called TopoFlow-IDL that supported multiple methods of modeling each hydrologic process and that also had a user-friendly graphical user interface (GUI). While ARHYTHM was written in Fortran and used triangular grid cells, TopoFlow was written in IDL (Interactive Data Language) and used rectangular grid cells with D8-based flow directions. The process treatments of snowmelt, evaporation, and shallow subsurface flow from ARHYTHM were converted to IDL, and array-based best programming practices (e.g., avoidance of spatial loops) were used to ensure good runtime performance. A wizard-style GUI for TopoFlow-IDL was developed, and from 2000 to 2007 support for many additional hydrologic processes were added. These additions can be summarized as follows:

Meteorology. The first major addition was a meteorology module based on celestial mechanics with shortwave and longwave radiation calculators, using the approach outlined in *Dingman* [2002] (Appendix E in the supporting information). This allowed the energy balance snowmelt and evaporation process modules to be used even when shortwave and longwave radiation measurements were unavailable. This module required measurements of precipitation rate, air temperature, relative humidity, and wind velocity as its primary inputs, to be read from files, but computed many output variables from these.

Infiltration. Three different methods for modeling infiltration were added, based on the work of *Smith et al.* [2002], namely the well-known Green-Ampt method, the Smith-Parlange three-parameter method, and the Richards equation (1-D) method.

Diversions. Support for flow diversions were added (e.g., canals and tunnels) as well as support for sources and sinks that add or remove some or all of the flow that passes through a given grid cell. Flow diversions are relatively common and must sometimes be modeled to properly account for mass balance. (e.g., the 37.3 km long Harold D. Roberts Tunnel under the Continental Divide from Dillon Reservoir to the South Platte River near Denver, Colorado). This capability has also been used to model flow in river delta distributaries [*Hannon et al.*, 2008].

D8 Toolkit. The D8 method [*Jenson*, 1985] allows flow direction (aspect) to be computed from a DEM. Given a grid of D8 flow direction codes, several additional geometric attributes can be computed including topographic slope, total contributing areas (TCA), and channel lengths. TCA can then be used to compute grid-based estimates of channel attributes including widths and roughness parameters as explained in *Peckham* [2009a]. This component contains all of the code necessary to compute D8-based attributes. (See detailed instructions in the supporting information appendices.)

All of the TopoFlow components are described in detail in *Peckham* [2009a] and also in HTML-based help files online that are listed in Appendix A in the supporting information. In addition to the hydrologic process components, TopoFlow-IDL includes a number of tools for preparing the input data required by the process components.

TopoFlow-IDL has been used in a number of studies, such as *Schramm* [2005], *Bolton* [2006], *Coe et al.* [2008], *Liljedahl* [2008], and *Pohl et al.* [2009], several of which involved Arctic hydrology.

2.2. The New, Component-Based, Python Version of TopoFlow

The NSF-funded Community Surface Dynamics Modeling System (CSDMS) project began in 2007, with Peckham serving as its Chief Software Architect until 2013. The CSDMS modeling framework was designed to support flexible, component-based, “plug-and-play” modeling and to promote reuse and coupling of open-source models written by different authors [*Peckham et al.*, 2013]. For component-based modeling, one has to decide on an appropriate level of *granularity*, that is, the level of functionality that components should encapsulate. In any given geoscience modeling context, there are typically multiple *physical processes* that contribute to conservation of mass, momentum, and energy, and there are typically multiple *methods* (from very simple to sophisticated) for modeling each process. Different methods may differ in various ways, such as (1) the set of equations and variables used to represent the process, (2) the numerical scheme for solving the equations, (3) the programming language, (4) the computational grid, or (5) the use of processors (e.g., parallel versus serial). Several examples of hydrologic processes were described in the previous section. It is therefore natural to encapsulate individual physical process treatments in interchangeable components, and this sets an appropriate and useful granularity for plug-and-play modeling.

While the original IDL version of TopoFlow already supported multiple methods for modeling each of several hydrologic processes—with users selecting methods from process droplists—all of these modules were dependent on the application to connect the selected modules. A user could only choose from modules that were already included in the application and could not easily replace some of them with modules written by others, nor use a TopoFlow process module outside of the TopoFlow-IDL app. This type of “all in one” model is typical, where the application basically serves as a self-contained framework that allows all of the process modules to interoperate.

As an efficient means of exploring different architectural designs to support plug-and-play modeling, most of the source code for TopoFlow-IDL was converted from IDL to Python/NumPy. TopoFlow was also decomposed into a set of independent process-level *model components* as shown in Figure 1. Each component then had its own time loop, its own configuration or CFG file (a text file read by the component at startup to set parameters, etc.), its own HTML help page, and so on. Any variables that needed to be passed between model components were made available through a standardized component interface (i.e., set of functions), and many different interface prototypes were tested against numerous design criteria. TopoFlow therefore became a vehicle for designing the CSDMS modeling framework [*Peckham et al.*, 2013], including the Basic Model Interface [*CSDMS-BMI*, 2016] and the CSDMS Standard Names [*Peckham*, 2014a].

The Basic Model Interface (BMI) enables simplified, plug-and-play reusability and coupling of model components, often written by different people, even when those models differ in terms of programming language, computational grid, time-stepping scheme, and the names, units, and data types of their input and output variables. The standardized set of functions in the BMI provide a modeling framework with (1) fine-grained control of model execution, (2) descriptive information needed for coupling, and (3) variable getter and setter functions to support exchanging the values of variables. Based on information retrieved from BMI functions, the modeling framework is able to automatically call its built-in *mediators* (e.g., regridders, unit converters,

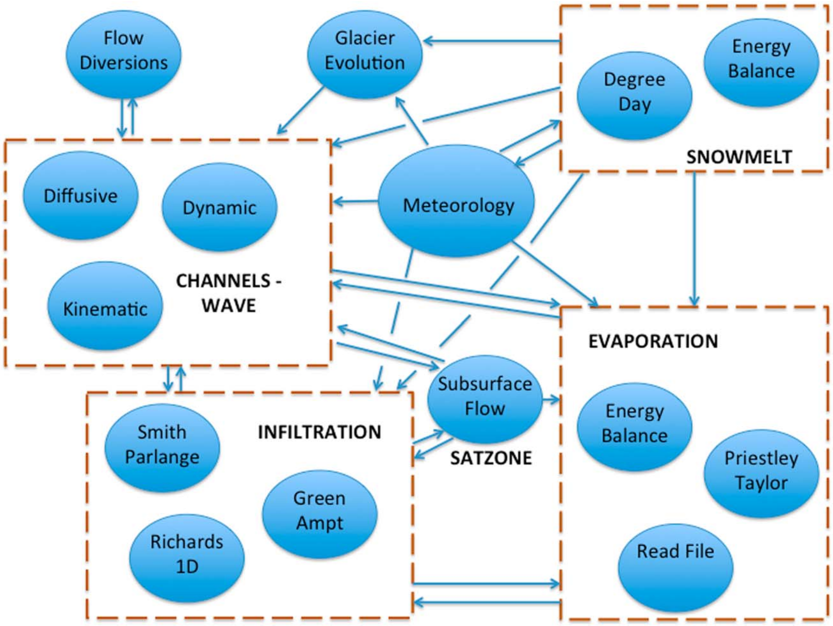


Figure 1. A diagram of all TopoFlow components, where arrows between components indicate dynamic coupling and the passing of one or more variables between components. Each component provides a method of modeling a particular hydrologic process, and the dashed boxes contain alternate methods for modeling the same process, ranging from simple to complex. Users choose one process component from each dashed box.

and time interpolators) to deal with differences between the coupled models. A unique feature of BMI is that it also addresses the problem of *semantic mediation* that arises from each model using its own set of names for input and output variables—its own *internal vocabulary*. A BMI implementation includes a simple mapping from a model's internal variable names to a set of standard variable names called the CSDMS Standard Names (CSNs). This mapping allows the framework to perform *automatic* semantic mediation. The CSNs are a systematic, unambiguous, cross-domain set of variable naming conventions which have evolved into a formal ontology called the Geoscience Standard Names [*GSN*, 2017].

Prior to 2012, CSDMS had developed a web-based graphical tool called the Component Modeling Tool (CMT) that CSDMS members could download as a lightweight Java application. CMT allowed users to select components from a palette and drag them into an arena to become part of a composite model. Once in the arena, CMT provided a tabbed-dialog GUI for *each model component* to collect the user settings necessary to automatically create a configuration file from a model-specific template. The GUI also provided standardized, HTML help pages for each component. Users could create new models from components, configure them, then run them on a high-performance cluster and also monitor their progress through the CMT. CMT has since been superseded by the CSDMS Web Modeling Tool (WMT), which is completely browser-based as well as more robust and efficient. TopoFlow was available in the CMT and is now available in the WMT, [*CSDMS-WMT*, 2016] by choosing *wmt-hydrology*. The WMT allows models to be composed and configured without logging into the high-performance cluster until they are ready for execution. Note that both CMT and WMT are simply graphical *front ends* to the CSDMS modeling framework, which itself is a software stack running on a cluster. They collect and store information in a modeling framework configuration file that is passed to the underlying framework. However, these configuration files may also be created with a text editor.

As a result of deconstructing TopoFlow into separate process-level model components, the Python version of TopoFlow required the presence of a model coupling framework like CSDMS in order to be used. However, some TopoFlow users wanted to be able to run TopoFlow on their own computers, without relying on a cluster with the full CSDMS software stack. This led to the development of a lightweight, experimental modeling framework written entirely in Python called EMELI (Experimental Modeling Environment for Linking and Interoperability) [*Peckham*, 2014b]. Users simply list the names of the model components they want to use in a *provider file*, and then EMELI 1.0 automatically connects each component to other components in the set that are able to provide the input variables they require, based on semantic matching with CSDMS Standard Names. EMELI requires each model component to have an implementation of BMI with Python bindings. EMELI also includes two mediators, a time interpolator and a unit converter, that are automatically invoked when model components use different time steps or units.

All of the TopoFlow model components, utilities, documentation, and a few example data sets are now available in a single, stand-alone Python package that is bundled with EMELI [*Peckham*, 2016, 2017]. Source code for EMELI is in the *framework* folder of this package. Source code for TopoFlow model components is in the *components* folder, and each has a BMI interface and uses CSDMS Standard Names. This includes a component called *d8_global.py* with a complete D8 toolkit for computing D8 flow direction grids and associated grids (e.g., total contributing area grids). Also included is a D8-based, fluvial landscape evolution model called Erode (*erode_d8_global.py*). Source code for a variety of shared, low-level TopoFlow utilities is in the *utils* folder, and these are used by all components. Some complete sets of input files for testing are included in the *examples* folder of the Python package. Altogether, the TopoFlow 3.5 package consists of over 71,000 lines of Python code, including comments. The TopoFlow 3.5 package has been tested on both the MacOS and Windows platforms, on top of an Anaconda Python distribution. It should also run on Linux platforms.

As part of an NSF EarthCube building block project called *GeoSemantics*, an alternate version of EMELI, *EMELI-Web* [2016], has been developed that can couple TopoFlow model components running on different servers as web services [*Jiang et al.*, 2017]. Values of variables that must be passed between components running on different servers are bundled in NetCDF files for transmission. It also provides a browser-based GUI, similar to WMT, for configuring each model component prior to a model run.

TopoFlow continues to be used in different contexts and in support of different cyber-infrastructure projects. For example, TopoFlow is used by CSDMS staff for teaching. Individual TopoFlow components can also be run within an iPython notebook, which makes it possible to follow every detail of model execution. TopoFlow is also being used in the NSF EarthCube building block project *OntoSoft* to help drive the development of standardized metadata and ontologies for describing geoscience models [*OntoSoft-CSDMS*, 2016].

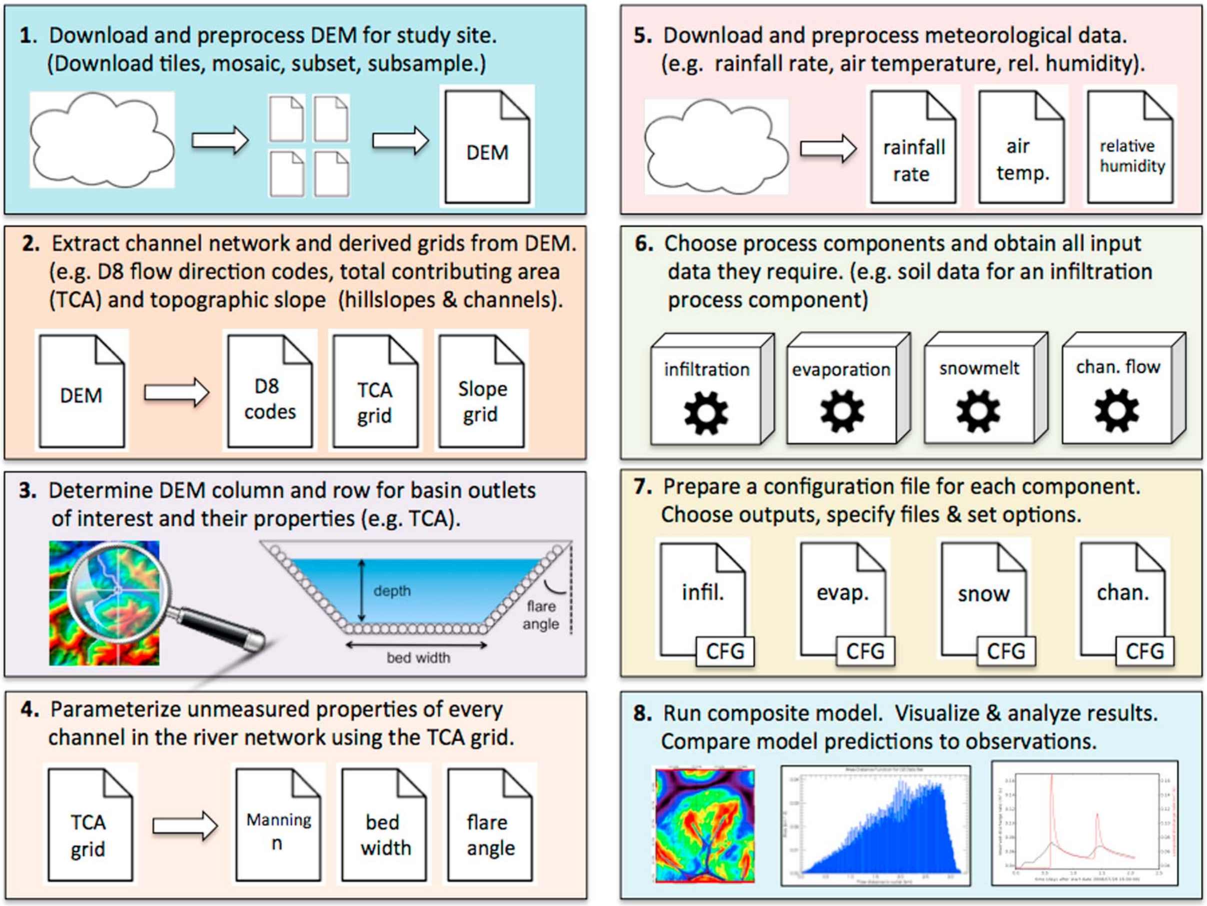
2.3. The Typical TopoFlow Workflow

With physically based, spatially distributed hydrologic models like TopoFlow, most of the work in using them has to do with acquiring and preparing the input data for the study site, configuring the model, and then running the model a large number of times with different parameter settings. (Without using a standard component interface like BMI, coupling process models can also be very time consuming.) Digital elevation data, which play a very important role in this type of modeling, are generally easy to find and download for any region of interest, at a variety of resolutions, thanks to the efforts of the U.S. Geological Survey and other agencies and projects. Many of the other types of input data, including meteorological, soil, and snowpack data, may not exist for a basin of interest, may have only been measured at a station some distance away, or may be difficult to find with an online search. However, this information is generally available online for experimental watersheds such as LTER (Long-Term Ecological Research) sites. Data at the scale of individual river channels, such as their widths, depths, and bed roughness, are generally not available and must be parameterized with known empirical relationships and limited field observations. Figure 2 provides a high-level illustration of the steps in a typical workflow. Details on how to perform steps 2 through 4 are included in multiple appendices in the supporting information. Note that steps 6 through 8 may be repeated many times with different choices of process components and different parameter settings in an effort to understand the physical processes at work in the basin and to make predictions that compare favorably with observations. For hydrologic modeling, one typically compares the predicted and observed *hydrographs* at the outlet of the basin of interest, which show the volume of water per unit time as a function of time (i.e., $Q(t)$) that flows through the outlet.

2.4. Preparing Input Files for TopoFlow

One of the unique features of TopoFlow is that, by design, almost any input variable that occurs in any of the process components can be provided as any of the following: (1) a *scalar* (a single value that does not vary in space or time), (2) a *time series* (a value that varies in time but not in space), (3) a *grid* (values that vary over space, but do not change over time), or (4) a *grid stack* (values that are variable in both space and time).

For example, a spatially uniform, steady rainfall rate would be provided as a scalar (to be used for all grid cells and all times), while a space-time rainfall field would be provided as a grid stack. A grid stack is essentially a time series of grids. Implementation of this feature is simplified because both IDL and Python are *dynamically*



1. Download and preprocess DEM for study site. (Download tiles, mosaic, subset, subsample.)

2. Extract channel network and derived grids from DEM. (e.g. D8 flow direction codes, total contributing area (TCA) and topographic slope (hillslopes & channels).

3. Determine DEM column and row for basin outlets of interest and their properties (e.g. TCA).

4. Parameterize unmeasured properties of every channel in the river network using the TCA grid.

5. Download and preprocess meteorological data. (e.g. rainfall rate, air temperature, rel. humidity).

6. Choose process components and obtain all input data they require. (e.g. soil data for an infiltration process component)

7. Prepare a configuration file for each component. Choose outputs, specify files & set options.

8. Run composite model. Visualize & analyze results. Compare model predictions to observations.

Figure 2. Typical steps in a TopoFlow modeling workflow.

typed programming languages, and *upcasting* therefore occurs automatically in expressions that contain a mixture of scalar values and grids. So functions in TopoFlow usually do not need to check whether arguments are scalar values or grids and can handle mixtures. Note that the term scalar is used in TopoFlow to mean *a single numerical value*; this could be confusing because a grid of values can be called a *scalar field*.)

For consistency, simplicity, and runtime performance, when TopoFlow reads the values of input variables from files, it expects (and requires) that (1) any time series is stored in a single-column text file (one value per line), (2) any grid is stored in an IEEE binary grid file, in row-major order, and (3) any grid stack is stored as a succession of binary grids, stored in a single file.

Binary grid files are a very common and computationally efficient file format. Appendix B in the supporting information describes binary grid files in more detail and explains how to read and write them with Python. Appendix C in the supporting information describes the binary grids that are needed to run TopoFlow. Appendix D in the supporting information explains how to prepare D8-based, binary grid input files for TopoFlow from a DEM with the D8 Global component. (The DEM is also assumed to be stored in a binary grid file.)

While all of the hydrologic process components in TopoFlow are optional and can be disabled with a setting in their configuration file, there are very few simulations that can be performed without using a channel flow component such as the *kinematic wave* component. (One example is snow depth accumulation using a snow component and meteorology component.) However, channel flow components require input files that describe both the channel geometry and bed roughness as spatial grids with the same dimensions as the DEM, as well as a D8 channel slope grid. This type of data is not readily available online, so TopoFlow users must prepare these grids. As explained in *Peckham* [2009a], these can be estimated by applying power law functions of the form $V = c\,(A + b)^p$ to a D8 total contributing area (TCA) grid, because both channel width and bed roughness vary with discharge (and therefore with TCA). However, the power law function parameters should be chosen carefully to obtain reasonable results. The IDL version of TopoFlow (TopoFlow-IDL) has a *Create* menu in its GUI that includes dialogs for computing these grids. However, one could also use Python commands similar to those in Appendices C and D in the supporting information to read a TCA grid into a variable, *A*, apply a power law function, and then write the resulting grid to a binary grid file. The compound filename extensions *_chan-w.rtg*, *_chan-a.rtg*, and *_chan-n.rtg* are commonly used in TopoFlow for grids of channel widths, channel trapezoidal bank angles (i.e., flare angles), and the channel roughness parameter, Manning's *n*, respectively.

Many of the other TopoFlow input variables represent either *initial conditions*—such as depth of water or snow, soil water content, and position of the water table—or *forcing variables* (or drivers), such as precipitation rate, air temperature, relative humidity, shortwave and longwave radiation fluxes, and wind speed [*Peckham*, 2009a]. These can be acquired from a variety of sources, including federal agencies, but must be either downloaded as or converted to binary grid files (or single-column text files for time series data). They may also need to be clipped or resampled to have the same dimensions and spatial extent as the DEM, which can be done using GIS software.

For output files, TopoFlow users can choose to save a time series, *profile series*, grid stack, or *cube stack* of data values to a NetCDF file or to binary grid files. A cube stack is a succession of 3-D arrays indexed by time, while a profile series is a succession of 1-D arrays indexed by time, such as a soil moisture profile that varies over time. These four types of output files include *0D*, *1D*, *2D*, or *3D* in their filenames, respectively. The 1-D and 3-D options are currently only used for subsurface flow variables. Users set flags in TopoFlow configuration files to choose which computed variables to save, using one of these four types of files. Often, one is interested in saving the values of some output variable (e.g., discharge, flow depth, or flow speed) at one or more specific locations (e.g., basin outlets) as a time series. This option can be turned on with a setting in a component's configuration file but requires the user to first create an *outlet file*—a simple text file with the following format:

```
-------------------------------------------------------------
 Monitored Grid Cell (Outlet) Information
-------------------------------------------------------------
  Column    Row   Area [km^2]   Relief [m]
-------------------------------------------------------------
    124    101    4.83503      385.0
```

The column and row values are essential and must be obtained with GIS software, such as RiverTools, but the area and relief values are nonessential. Multiple grid cells can be indicated for monitoring by adding rows.

Many applications provide a graphical user interface (GUI) that can make it much easier to prepare, process, edit, analyze, and visualize binary grid input files. For example, Appendix E in the supporting information explains how RiverTools and TopoFlow-IDL can be used. Additional information on preparing input files for TopoFlow can be found in *Peckham* [2009a] and in the TopoFlow Online Tutorial.

3. Example Application—Caribou-Poker Creek Research Watershed

3.1. Description of the Watershed

To illustrate the workflow of setting up and running TopoFlow, we use the data from the Caribou-Poker Creek Research Watershed (CPCRW) located 48 km north of Fairbanks 65° 10′ N, 147° 30′ W Alaska. The CPCRW site is part of the LTER (Long-Term Ecological Research) network. Parts of this watershed are underlain by permafrost, where the maximum seasonal thaw depth thickness is about 0.52 m at a low elevation point near the confluence of Poker and Caribou Creeks [*Bolton et al.*, 2000, 2004; *Bolton*, 2006]. Black spruce is generally found along poorly drained north facing slopes and valley bottoms. Aspen, birch, alder, and sporadic white spruce are found on the well-drained, south facing slopes. Tussock tundra, feather moss, and sphagnum mosses are also found along valley bottoms [*Bolton*, 2006]. The watershed encompasses an area of 101.5 km^2 as shown in Figure 3. CPCRW is located within the boreal forest area. The watershed has six subwatersheds, where three of them (C2, C3, and C4) have been continuously monitored over the last few decades. We chose to model the C2 subwatershed within the CPCRW because it is south facing and has almost no permafrost. South facing slopes usually correspond to a warmer microclimate and have a thinner organic layer and well-drained soils. The snowmelt at this site usually happens in a span of 1 or 2 weeks at the end of the spring season.

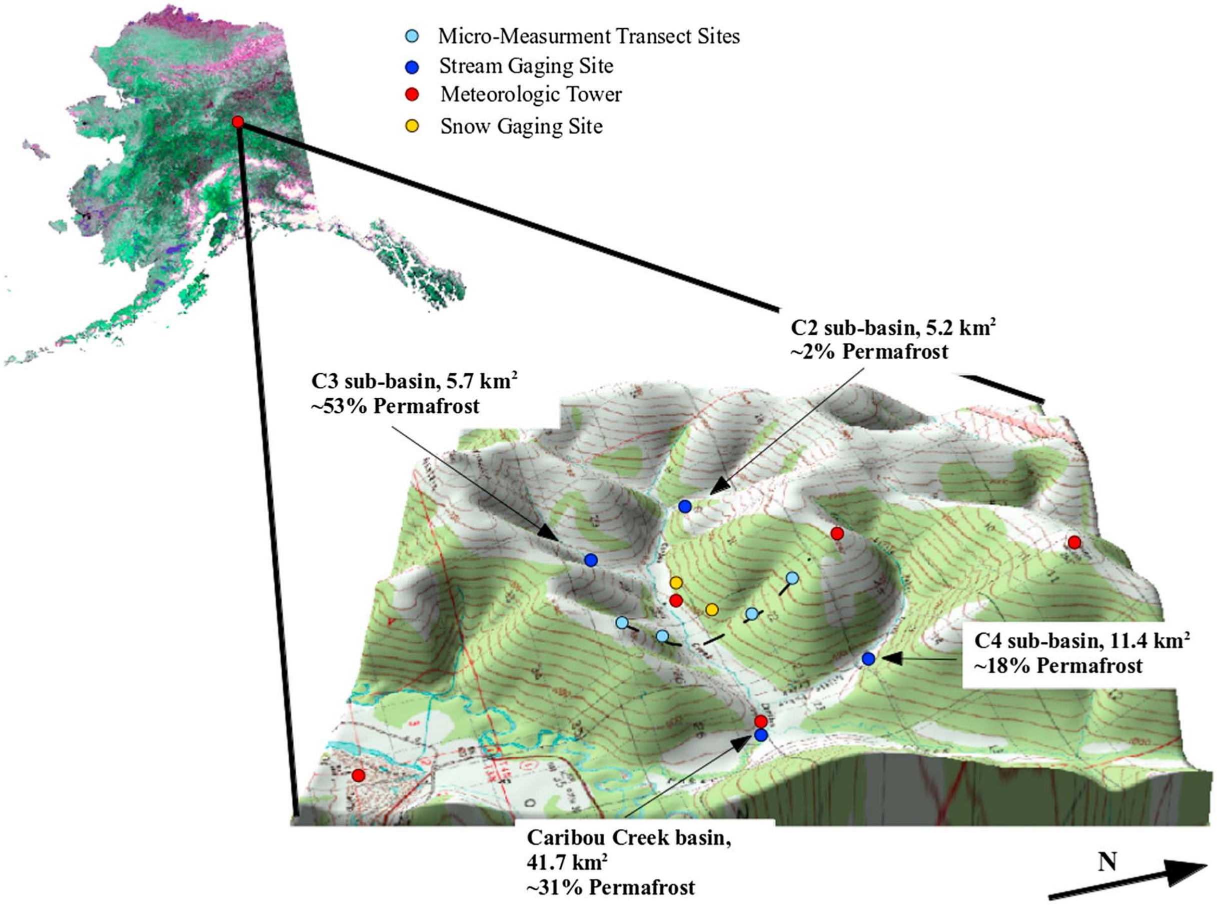


Figure 3. A map of the present measurement sensors in Caribou-Poker Creek Research Watershed study site.

Previous studies (e.g., *Bolton et al.* [2004]) have compared results for the C2 basin to those of the north facing, C3 basin, which has about the same basin area but has 54% permafrost versus 4% for C2. See Figure 4.

Figure 5 is a soil map for the vicinity of the C2 basin, clipped from a larger soil map contained in *Rieger et al.* [1972] (p. 14). It shows that the soil in the C2 basin is mostly Gilmore silt loam, with Fairplay silt loam near the drainage divide. Table 2 in *Rieger et al.* [1972] (p. 13) provides estimates of K_s (which they call *permeability*, as explained on p. 12), with values between 0.6 and 2.0 in/h or 4.23×10^{-6} to 1.41×10^{-5} m/s.

3.2. Acquiring a Digital Elevation Model for the Study Site

The TopoFlow model relies on a digital elevation model (DEM) in order to determine the flow directions and slopes that it uses to route water across the landscape. We used the following steps to obtain a DEM for our study site. While we think it is illustrative to describe these steps, we note that they are specific to a browser-based, graphical user interface (GUI) provided by the U.S. Geological Survey (USGS) as it exists at the time of writing — one that is likely to change in the future. While GUIs like this are typical and fairly easy to use, a hydrologist may need to use many different ones in order to acquire all of the input needed for a modeling study. This illustrates another challenge for reproducibility that is partially alleviated when data providers make their data available for download via a web service with a standardized API.

Step 1. The USGS provides an online tool called The National Map (TNM) for downloading data. In the navigation section on the left side of the page, check the box labeled *Elevation Products (3DEP)* in the *Data* section. This displays additional check boxes with different horizontal resolutions. We used 1 arcsec DEMs for this study, so we checked the box with this label. In the section labeled *File Format*, we checked the radio button labeled *GridFloat*. This is a simple, nonproprietary and efficient format that stores elevation values in row-major order (IEEE 4 byte, floating-point binary values), with the filename extension ".flt". Metadata — such as number of rows and columns, bounding box and grid cell dimensions — is saved in a small, companion text file with filename extension ".hdr".

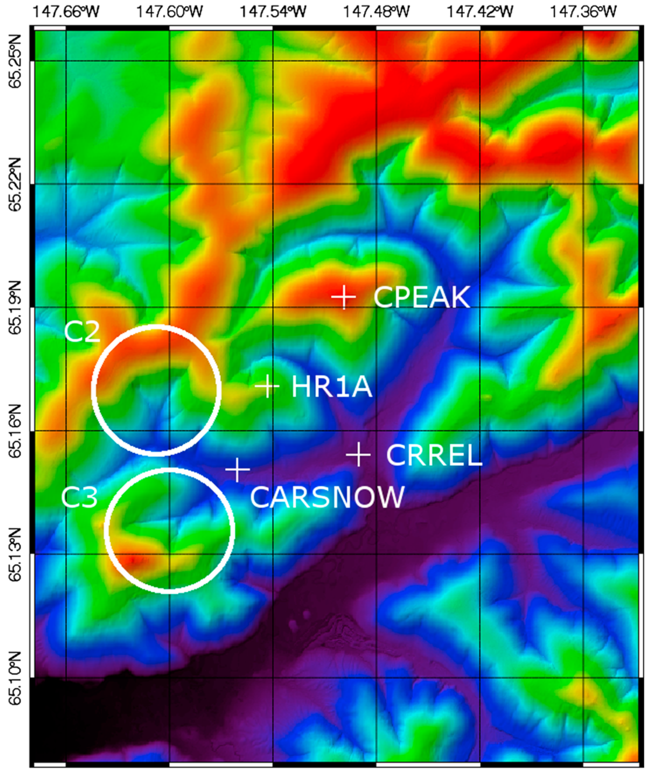


Figure 4. Locations of the C2 and C3 subbasins and the four met stations within the Caribou-Poker Creek Research Watershed. North is at the top of the image, and the circles have a diameter of about 3 km.

Step 2. Using the TNM browser-based map, we zoomed into the region labeled *Yukon Flats National Wildlife Refuge*, in the area southwest of (and downstream from) the confluence of the Porcupine River and the Yukon River. This is about 38 km north and slightly east of Fairbanks. The Caribou-Poker Creek Research Watershed is contained within the 1° by 1° tile with its southwest corner at 66° north latitude and 148° west longitude. We used the plus button to zoom into this region and clicked the radio button at the top labeled *Current Extent*. We then clicked on the blue button labeled *Find Products*. This resulted in a list of 1° DEM tiles, and we then selected the one labeled *USGS NED 1 arcsec n66w148 1×1 degree GridFloat 2016* by clicking on the shopping cart plus button to the right. The size of this file (one tile) is 42.99 MB. (A higher-resolution DEM for the same tile, with a grid spacing of 1/3 arcsec and file size of 377.98 MB, is also available.) Note that 1 arcsec of latitude is always roughly 93.6 m, while 1 arcsec of longitude decreases with latitude as 93.6 cos(lat) and is significantly less than 93.6 m for these Arctic latitudes.

3.3. Acquiring Data From the Bonanza Creek LTER Station

The Institute of Arctic Biology at the University of Alaska, Fairbanks maintains a Long-Term Ecological Research (LTER) site (funded by NSF) called the Bonanza Creek LTER. The Caribou-Poker Creeks Research Watershed (CPCRW) is one of the study sites for this LTER project where long-term monitoring data are collected and made available online. Clicking on *Access Data* > *Study Sites Catalog* in the *Data* menu of this website brings up a search filter page for the Study Sites Catalog. Typing *Caribou* in the text box labeled *Name, Description, History* and clicking on the *Submit* button generates a listing of available data and a locator map. For this paper, we selected the C2 subbasin within the Caribou Creek watershed. Data for four separate subbasins of Caribou Creek are available, namely C1, C2, C3, and C4. Note that each subbasin name begins with the letter “C” for Caribou. There are four weather stations (or “met stations”) in the vicinity of Caribou Creek, designated as CARSNOW, CPEAK, CRREL, and HR1A. The longitudes and latitudes of these stations are given by

```
CPEAK   -147.4990579, 65.19275149
CRREL   -147.4903787, 65.15425986
CARSNOW -147.5606703, 65.15065772
HR1A    -147.5435743, 65.17091866
```

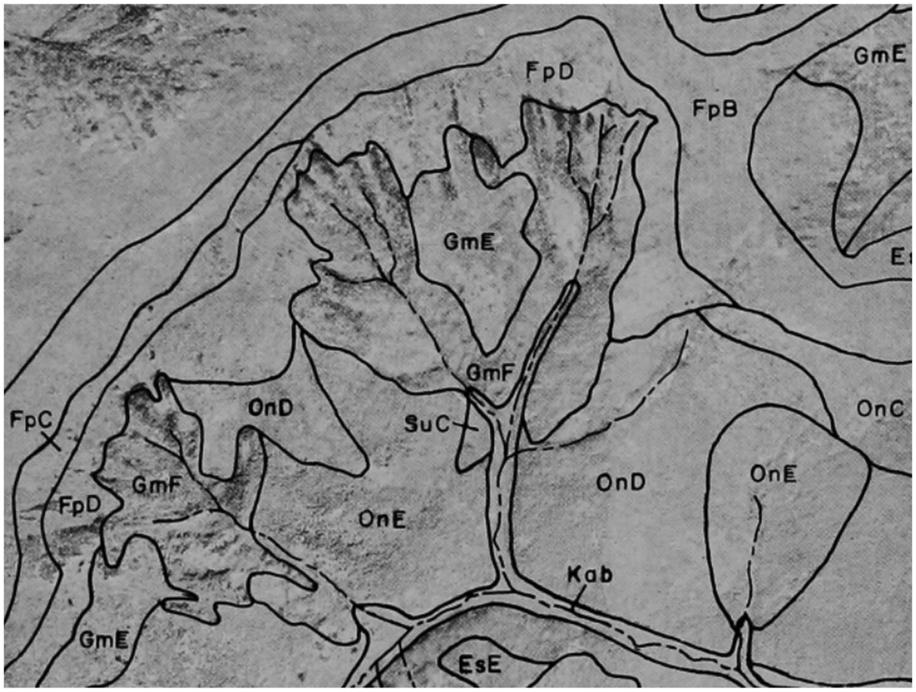


Figure 5. Soil map for part of the Caribou-Poker Creek Research Watershed that includes the C2 basin, from *Rieger et al.* [1972]. Soil types are all specific types of silt loam and corresponding slope ranges, as indicated with the following symbols: *Bradway*, Br; *Ester*: EsD (12 to 20%), EsE (20 to 30%), and EsF (30 to 40%); *Fairplay*: FpB (3 to 7%), FpC (7 to 12%), FpD (12 to 20%), and FpE (20 to 30%); *Gilmore*: GmD (12 to 20%), GmE (20 to 30%), and GmF (30 to 45%); *Karshner*: KaB (3 to 7%) and KaC (7 to 12%); *Olnes*: OnB (3 to 7%), OnC (7 to 12%), OnD (12 to 20%), OnE (20 to 30%), and OnF (30 to 45%); and *Saulich*: SuB (3 to 7%), SuC (7 to 12%), and SuD (12 to 20%).

Rainfall rates, measured with a tipping bucket, are stored in text files where the header and first line of data for the CARSNOW station look like this

```
site_id,date,hour,measurement,value,unit,flag
CARSNOW,2006-10-04,1400,Tipping Rain Bucket,0.000,mm,G
```

Similarly, discharge measurements for the C2, C3, and C4 subbasins are available in text files where the header and first line of data look like this

```
``Watershed'',``Date-Time'',``Flow'',``Units'',``Flag''
``C2'',7/14/1978 7:00:00,29.19,``L/s'',``G''
```

Volume flow rates (discharges) at the C2 basin outlet have been measured every summer since 1979. The measurement frequency was hourly until the summer of 2001, after which it was measured every 15 min. This long record presents an opportunity for investigating the possible effects of climate change. Additional metadata is available in a file called *README.rtf.doc* that comes with the data from the LTER website.

3.4. Preparing LTER Data for Model Use

Discharge and precipitation data were cleaned with the Python preprocessing scripts and notebooks that we have made available on GitHub [*Stoica*, 2016]. Version 1.0.0 of this GitHub repository was also archived and assigned a DOI via Zenodo-GitHub integration [*Stoica*, 2017]. The time-date format for both sets of data was converted to seconds with respect to a common reference date so that data could be properly aligned. Discharge data for the three basins (C2, C3, and C4) were interleaved, so we selected the desired data using the data frame manipulation utilities in the Pandas Python package. A sample of the data was selected based on the desired date-time interval values. Both data sets contained outliers that were mislabeled with "G" (to indicate that the data are good), so these were detected using histograms and then filtered out. Details and analysis of the data sets can be found at the GitHub link presented above.

3.5. Observed Response of the C2 Basin to a Late Summer Rainfall Event

Figure 6 shows rainfall rates that were measured with a tipping bucket at 1 h time intervals at the CPEAK met station for a rainfall event that took place over a 2 day period from 28 to 30 July 2008. Figure 7 shows

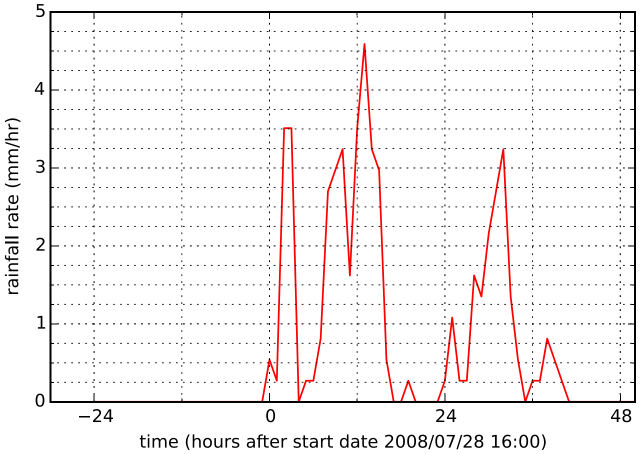


Figure 6. Measured rainfall rates at the CPEAK met station for a summer rainfall event: 28–30 July 2008.

the corresponding volume flow rates (discharges) measured at the outlet of the C2 basin. The C2 hydrograph shows that there is a base flow discharge of approximately 0.036m^3/s prior to the rainfall event. (In our model runs we start with dry channels and then add this amount of base flow to the resulting hydrograph.) The four main rainfall peak values in the measured rainfall time series for the July rainfall event at the CPEAK met station and the three corresponding peak discharges at the C2 basin outlet are given in Table 1. As shown in Figure 8, observed rainfall rates at the other met stations display a similar temporal intensity pattern, which supports treating rainfall as spatially uniform over the C2 basin as a first approximation. However, methods such as *inverse distance weighting* (IDW) could be used to create a grid sequence of rainfall rates to be used as input to TopoFlow. An IDW tool for this purpose is included with TopoFlow-IDL.

3.6. Model Component Selection and Setup

While TopoFlow includes numerous hydrologic process components, we deliberately chose to model a relatively simple situation where the dominant processes are surface flow, rainfall, and infiltration. We chose a late summer rainfall event so that contributions from snowmelt could be neglected. We also chose the C2 subbasin, which is south facing and almost free of permafrost. The near absence of permafrost, the relatively uniform soil type (Gilmore silt loam), and the selection of a rainfall event preceded by several days of no rainfall mean that the assumptions of the Green-Ampt infiltration model should be approximately satisfied. In addition, evaporation is considered to be negligible compared to the rainfall and infiltration rates. Finally, the relatively steep slopes throughout the C2 basin mean that the kinematic wave method of channel flow routing should be appropriate (i.e., no backwater effects, etc.) With these simplifications, we can focus on surface flow, as described by Manning's formula, and the infiltration process physics included in the Green-Ampt model.

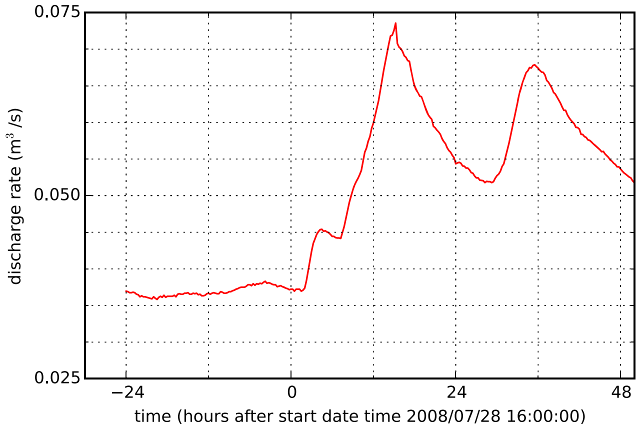


Figure 7. Volume flow rates measured at the outlet of the C2 basin for a summer rainfall event: 28–30 July 2008.

Table 1. Observed Peak Rainfall Rates at CPEAK and Corresponding Peak Discharges at the C2 Basin Outlet

Time (min)	Rainfall Rates (mm/h) ([m/s])	Time (min)	C2 Discharges (L/s)
120–240	$P_1 = 3.51$ (9.75×10^{-7})	255 – 285	$Q_1 = 45.4$
600–660	$P_{2a} = 3.24$ (9.00×10^{-7})	—	—
780–840	$P_{2b} = 4.59$ (1.28×10^{-6})	915 – 930	$Q_2 = 73.6$
1920–1980	$P_3 = 3.24$ (9.00×10^{-7})	2130 – 2145	$Q_3 = 67.9$

3.7. Choosing a Soil Water Retention Model for the Infiltration Process

In infiltration theory, there are four interrelated variables of interest that represent 3-D scalar fields below the land surface, namely K, the *hydraulic conductivity*, v, the *vertical component of the Darcy velocity*, θ, the *soil water content*, and ψ the *pressure head*. In order to create a mathematical model that can solve for these four variables, four equations are needed. Two equations are *conservation of mass* and *Darcy's Law*, and combining them results in the well-known *Richards equation* for modeling the flow of water through a porous medium (e.g., soil), driven by gravity as well as capillary suction. However, two additional equations are needed, and these are empirical relations of the form $K(\theta)$ and $\psi(\theta)$ known as *soil characteristic relations*, which allow K and ψ to be computed as functions of θ. *Brooks and Corey* [1964] proposed functional forms for $K(\theta)$ and $\psi(\theta)$ that depend on three parameters, namely $\psi_B < 0$, the *bubbling pressure*, and two model parameters η and λ, where $\eta = 2 + 3\,\lambda$. The parameters $\eta > 0$ and $\lambda > 0$ can be set to different values in order to provide good fits to observational data for the flow dynamics of different soil texture types. However, while their model has $K = K_s$ at saturation ($\theta = \theta_s$), it yields $\psi = \psi_B < 0$ instead of $\psi = 0$ at saturation. van Genuchten (1980) proposed an alternate pair of relations that also depend on three parameters, namely $\alpha_g < 0$, $m > 0$, and $n > 0$. While this model has $\psi = 0$ at saturation, its parameters are less physically meaningful and it has a complicated functional form for $K(\psi)$ [*Smith et al.*, 2002] (p. 21) that is difficult to integrate. (This makes it difficult to compute the parameter G in 8.)

Smith [1990] introduced a third pair of relations based on the Brooks-Corey model, which he called the *transitional Brooks-Corey* (TBC) model, which combines the benefits of the Brooks-Corey and van Genuchten models. *This is the soil water retention model that is used by every infiltration component in TopoFlow.* It introduces two new parameters $c > 0$ and ψ_A, such that the Brooks-Corey model is obtained in the limit as $c \to \infty$. For Smith's TBC model, the soil characteristic relations are given by

$$K\left(\Theta_e\right) = K_s\,\Theta_e^{\left(\frac{\eta}{\lambda}\right)} \tag{1}$$

$$\psi\left(\Theta_e\right) = \psi_B\left[\Theta_e^{\frac{-c}{\lambda}} - 1\right]^{\frac{1}{c}} - \psi_A. \tag{2}$$

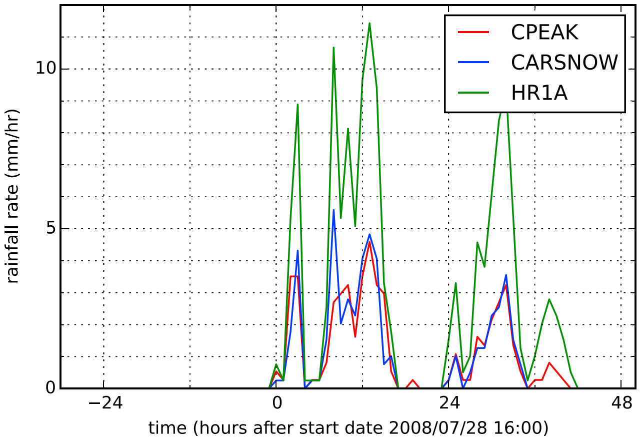


Figure 8. Measured rainfall rates at the CPEAK, CARSNOW, and HR1A met stations for a summer rainfall event: 28–30 July 2008.

At saturation, it has $K = K_s$ and $\psi = -\psi_A$, and one typically sets $\psi_A = 0$. The parameters η, λ, and c are not independent, and in fact $\eta = 2 + 3\,\lambda$ and $c = \eta/\lambda = 2/\lambda + 3$ in this model. Since $\lambda > 0$, this implies that $\eta > 2$ and $c > 3$. Here Θ_e is the *normalized soil water content* defined as

$$\Theta_e = \left(\frac{\theta - \theta_r}{\theta_s - \theta_r}\right), \tag{3}$$

and θ_r is the *residual water content*. Note that $\Theta_e(\theta_r) = 0$ and $\Theta_e(\theta_s) = 1$. For the TBC soil model, it is also possible to solve for $K(\psi)$ as

$$K(\psi) = K_s \left\{1 + \left[\left(\psi + \psi_A\right)/\psi_B\right]^c\right\}^{\left(\frac{-\eta}{c}\right)}. \tag{4}$$

3.8. Green-Ampt Infiltration Model Component—Theory

The Green-Ampt infiltration model [*Green and Ampt*, 1911; *Smith et al.*, 2002] can be derived as a physically based approximation to *Richards Equation*, which in turn is considered the best-available mathematical model for the process of infiltration. This approximation conserves the mass of water and incorporates a soil model (e.g., Brooks-Corey) but assumes that the initial soil moisture profile is uniform with depth and that lateral flow in the unsaturated zone can be neglected. It also assumes that there is a single, deep soil layer with uniform properties. Unlike the Richards equation model, the Green-Ampt model treats the variation of soil water content, θ, with depth below the land surface, z, as simple *piston flow* with a sharp wetting front—that is, with $\theta = \theta_s$ above the wetting front and $\theta = \theta_i$ below the wetting front.

Both the Green-Ampt and Smith-Parlange (three parameter) models of infiltration make use of what *Smith et al.* [2002] calls the *Infiltrability-Depth Approximation* or IDA. Instead of expressing the infiltration rate at the surface, v_0, as a function of time, t, the IDA instead expresses v_0 as a function of the *cumulative infiltration depth*, F, given by

$$F(t) = \int_0^t v_0(\tau)\,\mathrm{d}\tau. \tag{5}$$

Note that $v_0 = \mathrm{d}F/\mathrm{d}t$. This change of independent variable (from t to F) provides a more robust treatment of the relevant boundary conditions and the transition between them. According to the Green-Ampt model, v_0 is given by

$$v_0 = \min\left(P, f_c\right), \text{if } P > K_s \quad P, \text{if } P < K_s. \tag{6}$$

where f_c is called the infiltrability (or *infiltration capacity*) and represents the maximum possible infiltration rate that the soil will allow, given by

$$\begin{aligned} f_c &= K_i + \left[\left(K_s - K_i\right)\,(F + J)/F\right] \\ &= K_s + (J/F)\,\left(K_s - K_i\right). \end{aligned} \tag{7}$$

Here $J = G\left(\theta_s - \theta_i\right)$, and G is the *capillary length scale* [*Smith et al.*, 2002] defined as

$$G = \frac{1}{K_s}\int_{-\infty}^{0} K(\psi)\,\mathrm{d}\psi. \tag{8}$$

The Green-Ampt parameter, G, characterizes an initially dry soil with $K_i \ll K_s$.

For the TBC soil model used by TopoFlow, G can be computed by inserting (4) into (8), and for the typical case where $\psi_A = 0$, *Peckham* [2010] found the following closed-form expression for the resulting integral

$$G = -\psi_B\left[\frac{\Gamma(1 + 1/c)\,\Gamma\left[(\eta - 1)/c\right]}{\Gamma(\eta/c)}\right]. \tag{9}$$

Here $\Gamma(x)$ is the *Gamma function* and recall that $\psi_B < 0$. Since $\eta > 2$ and $c > 3$, it can be shown that $0 < G < -2\,\psi_B$. While *Smith et al.* [2002, p. 71] gave a closed-form expression for G in the case of the standard Brooks-Corey model, namely $G = -\psi_B\,\eta/(\eta - 1)$, this result for the TBC model appears to be new. It also yields the Brooks-Corey expression for G in the limit as $c \to \infty$, but can give very different values for smaller values of c. Note that while Mathematica has powerful symbolic integration capabilities and can evaluate the G integral for several specific values of c (e.g., $c = 1$, $c = 3/2$, $c = 2$), it does not provide the general expression in (9).

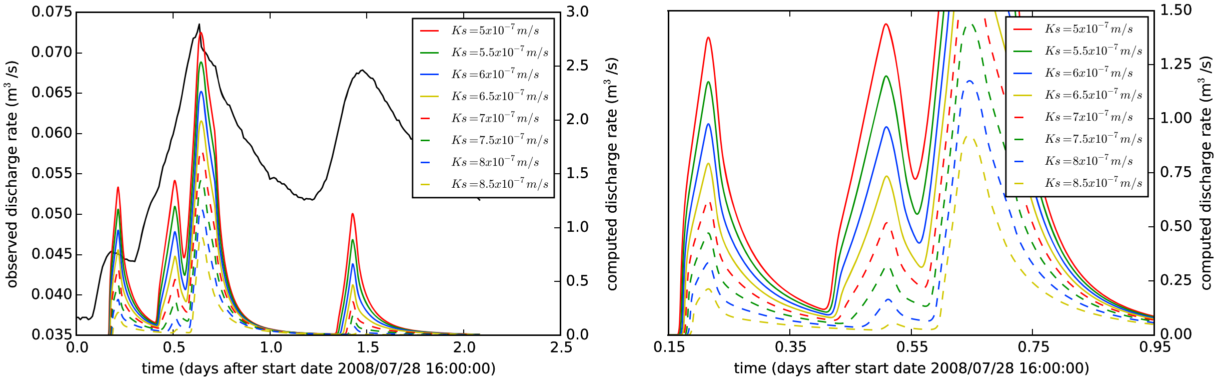


Figure 9. Effect of varying K_s with $\theta_i = \theta_s$ on precipitation peak response suppression. Notice that the observed and modeled hydrographs have different y axes.

Mathematica can also be used to check (9) by computing the G integral numerically for arbitrary choices of c and η.

3.9. Matching Rainfall Peaks to Hydrograph Peaks With Green-Ampt

Assuming no other gains from snowmelt or base flow, and no losses from evaporation, the runoff available to generate discharge is given by $R = (P - v_0)$, the difference between the rainfall rate and the surface infiltration rate. In the special case where the soil is saturated at the start of a rainfall event, we clearly have $K_i = K_s$, as well as $\theta_i = \theta_s$, in view of (1). Equation (7) then reduces to $f_c = K_s$ and equation (6) simplifies to

$$v_0 = K_s, \text{if } \; P > K_s \qquad P, \text{if } \; P < K_s. \tag{10}$$

In this case, the Green-Ampt model only allows rainfall rates to generate nonzero runoff when $K_s < P$, that is

$$R = P - K_s, \text{if } \; P > K_s \qquad 0, \text{if } \; P < K_s. \tag{11}$$

In the more general case when $K_i \ll K_s$, Green-Ampt predicts that the runoff rate associated with a given rainfall peak (at any point in the basin) should be given by (see (7))

$$R_k = \max\left\{ P_k - \left(K_s + \frac{C}{F_k} \right), 0 \right\}, \tag{12}$$

where $C = G\left(\theta_s - \theta_i\right)\left(K_s - K_i\right)$, and $F_k < F_{k+1}$ are the values of the cumulative infiltrated depth at the time a rainfall peak occurs. The fact that infiltration rates are higher at the beginning of a rainfall event (and F is a nondecreasing function) means that a rainfall peak of a given magnitude will be less strongly reflected in the basin hydrograph if it occurs toward the beginning of a rainfall event than if it instead occurs at a later time. This can be clearly seen in the C2 measured hydrograph, where the first and fourth rainfall peaks are comparable in magnitude but the first hydrograph peak is much smaller than the third.

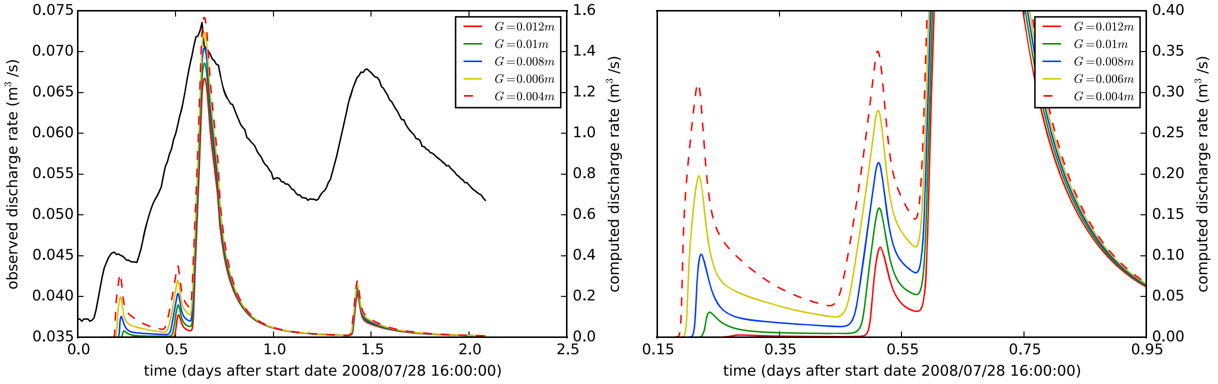


Figure 10. Effect of varying G with $K_s = 7.7 \times 10^{-7}$ m/s and $\theta_i = 0.01$ on precipitation peak response suppression. Notice that the observed and modeled hydrographs have different y axes.

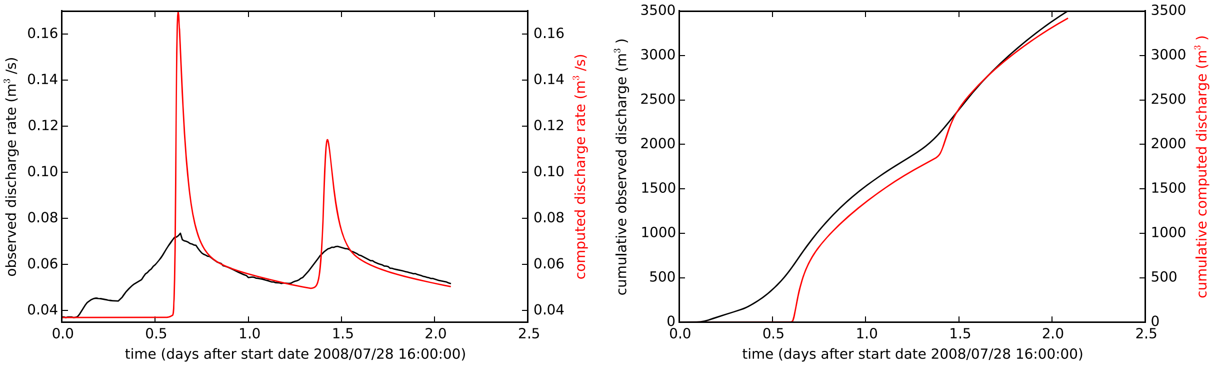


Figure 11. Best match result in terms of total volume discharge when $K_s = 4.5 \times 10^{-7}$ m/s, $\theta_i = 0.17$, and $G = 1.1$ m. The precipitation data were resampled for 2 s intervals using a uniform distribution, and the drag factor due to the Manning n coefficient was multiplied by a factor of 6.

In addition, the total contributing area of the C2 basin is $A = 4.84$ km^2 and the longest channel in the C2 basin is 2.93 km. Assuming a mean flow velocity of roughly 1 m/s in the channels, then once water reaches a channel, most water should reach the basin outlet in less than half an hour.

At a given point in the C2 basin, runoff will not occur unless the rainfall rate, P, exceeds the infiltrability, $f_c = K_s + C/F$; and therefore, it must exceed K_s. It follows that in order for the hydrograph peaks to have been generated by the corresponding rainfall peaks (although routed to the basin outlet by the channel network), we must have $P_1 > K_s$, $P_2 > K_s$, and $P_3 > K_s$. To satisfy all three inequalities, we must have $K_s < 9.0 \times 10^{-7}$ m/s. Note, however, that the rainfall rates were measured at 1 h intervals while the model time steps for both the infiltration and channel routing components were 2 s and the hydrograph at the outlet was measured at a time interval of 15 min. The actual rainfall peak values were likely higher (and no less than) than the recorded peak values. If the range estimate of $4.23 \times 10^{-6} < K_s < 1.41 \times 10^{-5}$ m/s given by *Rieger et al.* [1972] is accurate, then the instantaneous rainfall rates must have been higher than the low end of this range. But since these higher values were not measured, it would appear to be necessary to use values of K_s in the model at least *10 times smaller* in order to match the observed hydrograph.

3.10. Model Results and Analysis

We use equation (7) to guide our search for a good combination of parameters for the watershed. K_s is the offset that determines which pulses in the precipitation data are completely suppressed, since when $P < K_s$, the model indicates that all water is infiltrated. It is important to note, then, that the sampling rate for the precipitation, which in the case of this data set is 1 h, is critical to being able to utilize a K_s value that reflects the observed value. Since the precipitation sampling rate is low, we need to utilize a much lower K_s value than the reported value to preserve the peaks. Figure 9 shows the discharge at the tracked outlet in C2 for different values of K_s when $\theta_i = \theta_s$ so that $f_c = 0$. We notice that it is the second peak that is attenuated the most by an

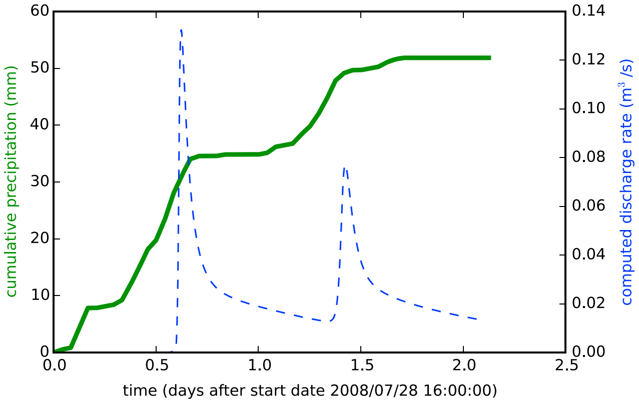


Figure 12. Best match result in terms of total volume discharge when $K_s = 4.5 \times 10^{-7}$ m/s, $\theta_i = 0.17$, and $G = 1.1$ m plotted against the cumulative precipitation.

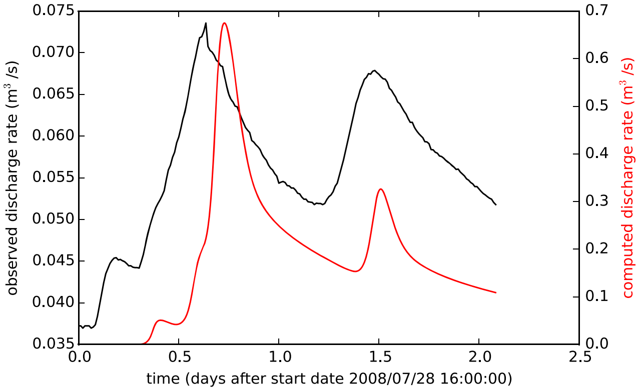


Figure 13. Illustrative example of the effect of increased surface drag (factor of 30 versus 6) on the curve shape of the modeled discharge. This model result yielded a volume discharge 10 times higher than observed but shows the features that could be captured for higher precipitation data sampling rates. Notice that the observed and modeled hydrographs have different *y* axes.

increase in K_s. Noting that there is a shoulder in the observed discharge output in Figure 7 indicates that the second peak is not attenuated and in fact looks to have approximately the same amplitude as the first peak. Therefore, selecting a K_s value for which a 1:1 ratio may be maintained between these two peaks is desirable; a K_s value above 8.0×10^{-7} seems to completely suppress the second peak and is thus not desirable.

For any given K_s value, introducing the second term in equation (7) by decreasing θ_i will further attenuate the peaks. The amount of attenuation and which peaks are attenuated most is determined by the weighting factor, *G*. Figure 10 illustrates how increasing *G* preferentially attenuates to first peaks more than latter peaks. The first three peaks all experience attenuation while the fourth peak is relatively unaffected.

With this understanding of parameter adjustment on modeled output, it is possible to find an optimum combination of parameters that will yield a reasonable output volume. Attempting to achieve the correct volume output while maintaining all the peaks results in incorrect peak response matching with the third peak being much larger than the other three peaks. Allowing suppression of the first two peaks and attempting to optimize the ratio of the third and fourth peak heights as well as the total volume discharge yields the result in Figure 11. In order to obtain this simulated output, several adjustments were made to the input data. First, the precipitation data were resampled at 2 s intervals (from the original 1 h), so that instead of a single pulse of precipitation at the beginning of each hour, the precipitation was added to the model in smaller increments. This resulted in a better overall precipitation volume estimate (within 100 m^3) in comparison to that modeled with the hour-interval precipitation (off by more than 6000 m^3). Additionally, the default surface drag effect from the Manning *n* coefficient was increased by a factor of 6. Without this drag effect, each pulse response occurs sooner and decays faster.

Figure 12 illustrates the way the model responds to a given input. It can be seen that the model behaves like a simple integrator upon being fed an impulse. It is the amount of cumulative precipitation during a short period of time that determines whether the model responds to a rain event, not the number of peaks during an event. Effectively, the precipitation event mimics a sequence of two step functions, and the outputs show how the motion of water through the landscape behaves similar to the discharging of a capacitor in an integrator circuit.

Lastly, in Figure 13, we show an example of the influence of increased surface drag and four-peak response on the simulated output. From this example, we can see the importance of these two effects in smoothing out the discharge curve.

A detailed, reproducible workflow of how the results in this section were obtained is available on GitHub as an iPython notebook [*Stoica*, 2016, 2017].

4. Conclusions and Recommendations

This paper has attempted to document the entire workflow of a spatial hydrologic modeling study to the extent that it can be easily reproduced and extended by other scientists. In support of reproducibility, the entire TopoFlow modeling toolkit, including components, utilities, and framework, is (1) open source and accessible on GitHub (MIT license); (2) version controlled; (3) easily installed as a Python package; (4) citable with a DOI [*Peckham*, 2017]; (5) extensible by adding new components; and (6) runnable with the CSDMS, EMELI, and EMELI-Web frameworks.

In addition, every TopoFlow model component has (1) object-oriented source code that takes advantage of inheritance; (2) a Basic Model Interface (BMI) to support plug-and-play reuse in frameworks; (3) all input and output variables mapped to CSDMS Standard Names; (4) additional, standardized metadata at OntoSoft portal (CSDMS section); (5) its own HTML help page that describes all variables and all equations used; (6) its own easy-to-read and edit configuration file (read at startup); (7) a graphical user interface (GUI): i.e., WMT for Python and TopoFlow-IDL for IDL; (8) the ability to run as a web service with EMELI-Web; and (9) the ability to save its output to standard format NetCDF files or generic binary grids. (These NetCDF files can be viewed with many visualization software toolkits, such as *VisIt* [2016].)

New components can easily be created by copying and then editing the Python source code of existing components. Components of a given process type can also inherit many capabilities from existing base classes. In addition, *BMI-to-Framework-X* adapters are under development by the EarthCube Earth System Bridge project that will allow components to run in several other modeling frameworks. These adapters are available in the *BMI-Forum* [2016] on GitHub.

All software used in this hydrologic modeling study has been made available in two GitHub repositories. The first one [*Peckham*, 2016] includes the complete TopoFlow 3.5 Python Package, which includes all documentation, components, utilities, the EMELI framework, and some example data sets for testing. The version 3.5.0 release has been archived and assigned a DOI with Zenodo-GitHub integration [*Peckham*, 2017]. The second one [*Stoica*, 2016] includes the complete set of Python scripts and iPython notebooks used in this specific study for (1) preprocessing the LTER data, (2) preparing CFG files for model runs, (3) plotting model output to create figures, and (4) analyzing model results. While the original LTER data files are too large for GitHub, direct links to these files can be found in the second GitHub repository. The DEM, D8-derived grids, CFG files, and other TopoFlow input files have been zipped and are also included in the second repository. Version 1.0.0 of this second repository was archived and assigned a DOI with Zenodo-GitHub integration [*Stoica*, 2017].

While used mainly as a vehicle for highlighting the issue of reproducibility and best practices, and while only tapping a small fraction of the capabilities of the TopoFlow model toolkit, the hydrologic modeling study for the C2 watershed led to interesting results that could be pursued further in several different directions. We showed how the features of a hydrograph resulting from a late summer storm could be interpreted using the Green-Ampt infiltration model. However, this study also clearly illustrated how insufficient temporal resolution in rainfall rate measurements suppresses the magnitudes of rainfall rate peaks, and how this prevents the model from matching observations unless parameters such as K_s are adjusted away from observed values. In this analysis, a robust, but lesser-known soil water retention model used by TopoFlow—known as transitional Brooks-Corey—was also highlighted, and a new, closed-form expression for G was provided.

Acknowledgments
We would like to thank Matt Nolan, who initiated the collaboration that led to TopoFlow, Larry Hinzman, who helped to guide and provide funding for its development, as well as the entire Arctic hydrology team at WERC (Water and Environmental Research Center), at the University of Alaska, Fairbanks. S. Peckham is the primary author of TopoFlow, all versions of which are free and open source, as well as the product RiverTools 4.0, which is sold commercially by Rivix, LLC. To avoid any perceived conflict of interest, this paper has described workflows that do not require RiverTools. This work was partially funded by NSF grants ICER 1440332 (GeoSoft), 1440333 (GeoSemantics), and PLR 1503559.

References

BMI-Forum (2016), Basic Model Interface (BMI) Forum on GitHub. [Available at https://github.com/bmi-forum.]

Bolton, W. R. (2006), Dynamic modeling of the hydrologic processes in areas of discontinuous permafrost, PhD thesis, 163 pp., Univ. of Alaska, Fairbanks, Dept. of Civil Engineering.

Bolton, W. R., L. D. Hinzman, and K. Yoshikawa (2000), Stream flow studies in a watershed underlain by discontinuous permafrost, in *Proceedings, Water Resources in Extreme Environments*, edited by D. L. Kane, pp. 31–36, Am. Water Resour. Assoc., Anchorage, Alaska.

Bolton, W. R., L. D. Hinzman, and K. Yoshikawa (2004), Water balance dynamics of three small catchments in a sub-arctic boreal forest, in *Northern Research Basins Water Balance Workshop Proceedings*, vol. 290, edited by D. L. Kane and D. Yang, pp. 213–223, IAHS Publication, Victoria, Canada.

Brooks, R., and A. Corey, (1964), Hydraulic properties of porous media, Hydrology Papers, Colorado State Univ., Fort Collins, Colo.

Coe, J. A., D. A. Kinner, and J. W. Godt (2008), Initiation conditions for debris flows generated by runoff at Chalk Cliffs, central Colorado, *Geomorphology*, *96*, 270–297, doi:10.1016/j.geomorph.2007.03.017.

CSDMS-BMI (2016), The Basic Model Interface (BMI). Online documentation, CSDMS. [Available at http://bmi-python.readthedocs.io/en/latest/.]

CSDMS-WMT (2016), Web Modeling Tool (WMT). [Available at https://csdms.colorado.edu/wmt/.]

Dingman, S. L. (2002), *Physical Hydrology*, 2nd ed., 646 pp., Prentice Hall, Upper Saddle River, N. J.

Edwards, P. N. (2010), *A Vast Machine: Computer Models, Climate Data, and the Politics of Global Warming*, MIT Press, Cambridge, Mass.

Edwards, P. N., M. S. Mayernick, A. L. Batcheller, G. C. Bowker, and C. L. Borgman (2011), Science friction: Data, metadata, and collaboration, *Soc. Studies Sci.*, *41*, 667, doi:10.1177/0306312711413314.

EMELI-Web (2016), Experimental Modeling Environment for Linking and Interoperability, Web service version. [Available at http://ecgs.ncsa.illinois.edu/emeli-web/.]

Garijo, D., P. Alper, K. Belhajjame, O. Corcho, Y. Gil, and C. Goble (2013), Common motifs in scientific workflows: An empirical analysis, *Future Gener. Comput. Syst.*, *36*, 338–351, doi:10.1016/j.future.2013.09.018.

Green, W. H., and G. A. Ampt (1911), Studies on soil physics: Part I. The flow of air and water through soils, *J. Agric. Sci.*, *4*(1), 1–24.

GSN (2017), Geoscience Standard Names (GSN) ontology. [Available at http://www.geostandardnames.org.]

Hannon, M. T., J. P. M. Syvitski, and A. J. Kettner (2008), Hydrologic modeling of a tropical river delta by applying remote sensing data: The Niger Delta and its distributaries, *Eos Trans. AGU*, *89*(53), Fall Meet. Suppl., Abstract H53B–1050.

Henderson, F. M. (1966), *Open Channel Flow*, Macmillan Publishing Co., New York.

Hinzman, L. D., D. L. K. DL, C. Benson, and K. Everett (1996), Chapter 6: Energy balance and hydrological processes in an Arctic watershed, in *Landscape Function and Disturbance in Arctic Tundra, Ecological Studies*, vol. 20, edited by J. Reynolds and J. Tenhunen, chap. 6, pp. 131–154, Springer-Verlag, Berlin, doi:10.1007/978-3-662-01145-4_6.

Hinzman, L. D., D. Goering, and D. L. Kane (1998), A distributed thermal model for calculating temperature profiles and depth of thaw in permafrost regions, *J. Geophys. Res.*, *103*(D22), 28,975–28,991.

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

IDL VM (2016), Interactive Data Language (IDL) Virtual Machine. [Available at http://www.harrisgeospatial.com/Support/HelpArticlesDetail/TabId/219/ArtMID/900/ArticleID/12395/The-IDL-Virtual-Machine.aspx.]

IEEE (2008), IEEE 754-2008, Institute of Electrical and Electronics Engineers. [Available at https://standards.ieee.org/findstds/standard/754-2008.html.]

Jenson, S. K. (1985), Automated derivation of hydrologic basin characteristics from digital elevation model data, in *Proceedings of the Digital Representations of Spatial Knowledge*, pp. 301–310, Auto-Carto VII, Washington, D. C. [Available at http://mapcontext.com/autocarto/proceedings/auto-carto-7/, accessed 2017-16-05.]

Jiang, P., M. Elag, P. Kumar, S. D. Peckham, L. Marini, and L. Rui (2017), A service-oriented architecture for coupling web service models using the Basic Model Interface (BMI), *Environ. Modell. Softw.*, *92*, 107–118. [Available at https://doi.org/10.1016/j.envsoft.2017.01.021.]

Liljedahl, A. (2008), Master's thesis, Univ. of Alaska, Fairbanks, Alaska.

OntoSoft-CSDMS (2016), OntoSoft software repository for CSDMS. [Available at http://csdms.ontosoft.org/#list.]

Parlange, J.-Y., I. G. Lisle, R. D. Braddock, and R. E. Smith (1982), The three-parameter infiltration equation, *Soil Sci.*, *133*(6), 337–341.

Peckham, S. D. (2009a), Chapter 25: Geomorphometry and spatial hydrologic modelling, in *Geomorphometry: Concepts, Software, Applications, Developments in Soil Science*, vol. 33, edited by S. D. Peckham, pp. 579–602, Elsevier, doi:10.1016/S0166-2481(08)00025-1.

Peckham, S. D. (2009b), Chapter 18: Geomorphometry in RiverTools, in *Geomorphometry: Concepts, Software, Applications, Developments in Soil Science*, vol. 33, edited by S. D. Peckham, pp. 411–430, Elsevier, doi:10.1016/S0166-2481(08)00018-4.

Peckham, S. D. (2010), TopoFlow soil properties page. [Available at https://csdms.colorado.edu/wiki/Model_help: TopoFlow-Soil_Properties_Page.]

Peckham, S. D. (2014a), The CSDMS Standard Names: Cross-domain naming conventions for describing process models, data sets and their associated variables, in *Proceedings of the 7th International Congress on Environmental Modelling and Software*, edited by D. P. Ames, N. W. T. Quinn, and A. E. Rizzoli, pp. 12, International Environmental Modelling and Software Society (iEMSs), San Diego, Calif. [Available at http://scholarsarchive.byu.edu/iemssconference/2014/Stream-A/12/, accessed 2017-16-05.]

Peckham, S. D. (2014b), EMELI 1.0: An experimental smart modeling framework for automatic coupling of self-describing models, in *Proceedings of HIC 2014: 11th International Conference on Hydroinformatics*, CUNY Acad. Works, New York. [Available at http://academicworks.cuny.edu/cc_conf_hic/464/, accessed 2017-16-05.]

Peckham, S. D. (2016), TopoFlow Python package on GitHub, open-source. [Available at https://github.com/peckhams/topoflow.]

Peckham, S. D. (2017), TopoFlow 3.5 Python package, peckhams/topoflow. [Available at http://doi.org/10.5281/zenodo.322649.]

Peckham, S. D., and J. L. Goodall (2013), Driving plug-and-play models with data from web-services: A demonstration of interoperability between CSDMS and CUAHSI-HIS, *Comput. Geosci.*, *53*, 154–161, doi:10.1016/j.cageo.2012.04.019.

Peckham, S. D., E. W. H. Hutton, and B. Norris (2013), A component-based approach to integrated modeling in the geosciences: The design of CSDMS, *Comput. Geosci.*, *53*, 3–12.

Pohl, S., P. Marsh, C. Onclin, and M. Russell (2009), The summer hydrology of a small upland tundra thaw lake: Implications to lake drainage, *Hydrol. Processes*, *23*, 2536–2546.

Priestley, C. H. B., and R. J. Taylor (1972), On the assessment of surface heat flux and evaporation using large-scale parameters, *Mon. Weather Rev.*, *100*(2), 81–92.

Richards, L. A. (1931), Capillary conduction of liquids through porous mediums, *J. Appl. Phys.*, *1*(5), 318–333, doi:10.1063/1.1745010.

Rieger, S., C. E. Furbush, D. B. Schoephorster, H. Summerfield Jr., and L. C. Geiger, (1972), Soils of the Caribou-Poker Creeks Research Watershed, Interior Alaska, Tech. Rep. 236, U.S. Army Corps of Engineers, Cold Regions Research and Engineering Lab (CRREL), Hanover, New Hampshire, Department of Agriculture, Soil Conservation Service.

RiverTools (2016), RiverTools Home Page, Rivix Software LLC. [Available at http://www.rivertools.com.]

Schramm, I. (2005), Hydrologic modeling of an arctic watershed, Alaska, PhD thesis, Univ. of Potsdam, Germany.

Smith, R. E. (1990), Analysis of infiltration through a two-layer soil profile, *Soil Sci. Soc. Am. J.*, *54*(5), 1219–1227.

Smith, R. E., and J.-Y. Parlange (1978), A parameter-efficient hydrologic infiltration model, *Water Resour. Res.*, *14*(3), 533–538.

Smith, R. E., K. R. J. Smettem, P. Broadbridge, and D. A. Woolhiser (2002), *Infiltration Theory for Hydrologic Applications*, Water Resources Monograph Series, vol. 15, 212 pp., AGU, Washington, D. C.

Stoica, M. (2016), Python processing scripts and notebooks on GitHub. [Available at https://github.com/mariutzica/Paper-of-the-Future-Content.]

Stoica, M. (2017), First release of topoflow demo for potf content, mariutzica/paper-of-the-future-content [data set]. [Available at http://doi.org/10.5281/zenodo.345140.]

Vardi, M. Y. (2010), Science has only two legs, *Commun. ACM*, *53*(9), 5, doi:10.1145/1810891.1810892.

VisIt (2016), Visit visualization software, Lawrence Livermore National Laboratory. [Available at https://wci.llnl.gov/simulation/computer-codes/visit.]

Zhang, Z., D. L. Kane, and L. D. Hinzman (2000), Development and application of a spatially-distributed arctic hydrological and thermal process model (ARHYTHM), *Hydrol. Processes*, *14*(6), 1017–1044, doi:10.1002/(SICI)1099-1085(20000430)14:6<1017::AID-HYP982>3.0.CO;2-G.