Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2017-321
Manuscript under review for journal Geosci. Model Dev.
Discussion started: 30 January 2018

Geoscientific Model Development Discussions
Open Access

GSFLOW-GRASS v1.0.0: GIS-enabled hydrologic modeling of coupled groundwater–surface-water systems

G.-H. Crystal Ng[1,2], Andrew D. Wickert[1,2], Lauren D. Somers[3], Leila Saberi[1], Collin Cronkite-Ratcliff[4], Richard G. Niswonger[5], and Jeffrey M. McKenzie[3]

[1]Department of Earth Sciences, University of Minnesota, Minneapolis, Minnesota, USA
[2]Saint Anthony Falls Laboratory, University of Minnesota, Minneapolis, Minnesota, USA
[3]Department of Earth and Planetary Sciences, McGill University, Montreal, Quebec, Canada
[4]Geology, Minerals, Energy and Geophysics Science Center, United States Geological Survey, Menlo Park, California USA
[5]Earth Systems Modeling Branch, United States Geological Survey, Menlo Park, California, USA

Correspondence to: G.-H. Crystal Ng (gcng@umn.edu)

Abstract.

Water flow through catchments sustains ecosystems and human activity, shapes landscapes, and links climate to the outermost layers of the solid Earth. The profound importance of water moving between the atmosphere and aquifers has led to efforts to develop and maintain coupled models of surface water and groundwater. However, developing inputs to these models is usually time-consuming and requires extensive knowledge of software engineering, often prohibiting their use by many researchers and water managers, and thus reducing these models' potential to promote science-driven decision-making in an era of global change and increasing water-resource stress. In response to this need, we have developed GSFLOW-GRASS, a straightforward set of open-source tools that develops inputs for and runs GSFLOW, the U.S. Geological Survey's coupled groundwater–surface-water flow model. As inputs, GSFLOW-GRASS requires at a minimum a digital elevation model, a precipitation and temperature record, and estimates of channel parameters and hydraulic conductivity. GSFLOW-GRASS is written in Python as a set of (1) GRASS GIS extensions, (2) input-file-builder scripts, and (3) visualization scripts. We developed a set of custom GRASS GIS commands that generate "hydrologic response units" for surface water, discretized topologically as sub-basins of the tributary network; build the MODFLOW grid; and add necessary attributes to each of these geospatial units. These GIS outputs are interpreted by a second set of Python scripts, which link them to hydrologic variables, build inputs to GSFLOW, and run GSFLOW. Lastly, GSFLOW output files are used to produce figures and time-lapse movies of simulation results using a third set of post-processing Python scripts. We demonstrate the broad applicability of these tools to diverse settings through examples based on: the high Peruvian Andes, the Channel Islands of California, and the formerly-glaciated Upper Mississippi valley in Minnesota .

1 Introduction

Predicting and understanding the hydrologic impacts of climate, land-use, and other natural and anthropogenic change is a scientific endeavor that is increasingly necessary to manage water resources. Addressing this need requires streamlined access to models that integrate surface and subsurface processes across a watershed. This integrated approach is required because traditional hydrologic models that focus only on a single component within a watershed cannot properly predict the effects of changing conditions and feedbacks across their boundaries.

Driven by the growing recognition of tightly coupled groundwater and surface water dynamics and the need to evaluate and manage the two as a single resource (Winter et al., 1998), the United States Geological Survey (USGS) developed and released GSFLOW. This integrated hydrologic model couples the groundwater flow model MODFLOW with the rainfall–runoff model PRMS (Precipitation Runoff Modeling System) (Markstrom et al., 2008). Both MODFLOW (Harbaugh, 2005; Niswonger et al., 2011) and PRMS (Leavesley et al., 1983; Markstrom et al., 2015) are popular models with significant user bases. GSFLOW has been applied to various watersheds in the US, for example in California (Essaid and Hill, 2014), Wisconsin (Hunt et al., 2013), Pennsylvania (Galeone et al., 2016), and Oregon (Surfleet and Tullos, 2013; Gannett et al., 2017), as well as applications outside of the US (e.g., Hassan et al., 2014; Tian et al., 2015).

Although the USGS completed the software development work required to fuse MODFLOW and PRMS into a single model, the process of implementing GSFLOW includes many hurdles that require significant time and computational knowledge to overcome (Gardner et al., 2017). To run GSFLOW, the user must first generate multiple formatted ASCII files corresponding to the individual MODFLOW and PRMS model inputs, along with an additional GSFLOW-specific file containing information about the alignment of these two single-domain models. Freely available USGS GUIs – ModelMuse (Winston, 2009) and the PRMS GUI (Markstrom et al., 2015) – and proprietary GUIs (mostly for MODFLOW) can help users develop inputs to each of these models. However, with the exception of a single for-fee software product by Earthfx (http://www.earthfx.com/), these tools neither help users conceptually link the different process domains represented in PRMS and MODFLOW nor generate the files that link them. The expertise required to develop inputs to GSFLOW are not often available at research institutes or management organizations, and there is a potential to significantly increase GSFLOW's user base. This lack of support software for developing GSFLOW models, and support software for integrated models in general, motivates our present work and we hope will enable more widespread hydrologic modeling.

Here we offer a free, platform-flexible, and easy-to-use utility tool that automatically builds all input files needed for GSFLOW. This tool consists of a domain-building module that generates and links a network of stream segments and watershed sub-basins to a MODFLOW grid; a set of Python scripts that create self-consistent GSFLOW, PRMS, and MODFLOW input files; and a final post-processing set of Python scripts for visualizing simulation results. The domain-building module requires a digital elevation model (DEM) as input, and uses GRASS GIS, an open-source GIS platform to build topographically defined sub-watersheds as the surface-water (PRMS) discretization. GRASS GIS is computationally efficient and offers a wide range of raster and vector algorithms. Use of open-source software facilitates implementation of the GSFLOW-GRASS by diverse academic, government, and individual entities, enables further community development of GSFLOW-GRASS, and aligns with

Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2017-321
Manuscript under review for journal Geosci. Model Dev.
Discussion started: 30 January 2018

the USGS's goal to make its resources universally accessible. Our effort complements ongoing USGS development of an ArcGIS-based input file creator that employs overlapping rectangular grids for both MODFLOW and PRMS (Gardner et al., 2017), in contrast to topologically based surface units created in GSFLOW-GRASS.

Together with the new software package by Gardner et al. (2017), GSFLOW-GRASS will equip the hydrologic modeling community for tackling open questions about the relative advantages of regular gridded HRU's versus unstructured HRU's for both research and resource management needs. Major model intercomparison projects have included representatives from each category (Reed et al., 2004; Maxwell et al., 2014). Gridded domains are easier to construct and distribute over parallel computational architectures, and they allow flexible spatial specification of soil and land-cover heterogeneity. In contrast, ungridded domains can conform better to complex terrain. Models such as tRIBS (Vivoni et al., 2004) and PIHM (Qu and Duffy, 2007) utilize triangulated irregular networks for better water balance performance over steep catchments. Similar to GSFLOW-GRASS, other hydrological models with ungridded domains use topographically defined sub-basins as efficient computational units, including SWAT (Arnold and Fohrer, 2005), SAC-SMA (Ajami et al., 2004), HEC-HMS (Feldman, 2000), and TOPNET (Bandaragoda et al., 2004). In addition to water balance and computational efficiency motivations, use of sub-basins in GSFLOW-GRASS further facilitates linkages with network-theory-based mappings of water and sediment transport (e.g., Czuba and Foufoula-Georgiou, 2014, 2015) and their ecological impacts (e.g., Hansen et al., 2016). Because of GSFLOW's physical-conceptual water-routing scheme (versus fully physical), numerical differences between sub-basin versus gridded HRU's are difficult to predict, but new automated toolkits offered here and by Gardner et al. (2017) will enable rigorous testing and allow users to choose the option that best suits their particular application.

2 Background

2.1 GSFLOW

GSFLOW simulates spatially distributed surface to subsurface water flow in a watershed using modified model codes from PRMS and MODFLOW. Although GSFLOW can run in modes equivalent to the stand-alone PRMS-IV model and the stand-alone MODFLOW model, only the "integrated" coupled version is described here. Near-surface watershed processes within the shallow "soil zone," including evapotranspiration, infiltration, runoff, and interflow are represented by the PRMS sub-component of GSFLOW. Groundwater flow below the "soil zone," including vertical soil water movement in the deeper unsaturated zone and saturated flow through horizontal aquifer layers, is represented by MODFLOW. Streamflow and exchange between streams and underlying groundwater systems are also represented by the MODFLOW sub-component. We describe here the key features of GSFLOW in order to guide new users in implementing it and interpreting its results; Markstrom et al. (2008) documents the full details.

2.1.1 Domain discretization

GSFLOW adopts a hybrid spatial domain discretization approach (Figure 1) to establish its computational units. Stream segments are links in a river network that are used in both the PRMS and MODFLOW sub-components of GSFLOW (Figure 1A). Horizontally, the PRMS sub-component uses hydrological response units (HRUs), its fundamental discretized unit, of any shape (Figure 1B). These are used for calculations of the upper soil zone and the part of the surface not covered by the stream network. The MODFLOW sub-component uses rectangular grid cells for the deeper subsurface (Figure 1C) and to further discretize the stream network into reaches (Figure 1D). Establishing reaches as the fundamental unit of computation for the stream network instead of segments makes it possible to resolve fine spatial resolution groundwater-surface exchanges. Like MODFLOW grid cells, HRUs can be set to rectangles, but they are also commonly defined topologically to correspond to sub-basins, as they are in our approach (Figure 1). An extension to this could include further subdividing sub-basins according to vegetation, soil type, or other geographic features to produce HRUs.

Vertically, the PRMS sub-component is discretized into conceptual shallow soil zone reservoirs, which do not correspond directly to physical locations within the soil column but are instead based on user-specified conceptual thresholds. Specifically, within an HRU, the “soil zone” is subdivided into three reservoir types – the capillary reservoir, gravity reservoir, and preferential-flow reservoir, which are filled in order of increasing water storage using efficient water accounting calculations (section 2.1.2) (Figure 2). Underlying the PRMS soil zone are MODFLOW grid cells representing the deeper unsaturated zone and the saturated zone. While grid cells have uniform horizontal discretization, vertical layer thicknesses can be variable in order to accommodate different hydrostratigraphy. To link the PRMS and MODFLOW grids, the user must define gravity reservoirs at each different intersection of an HRU and a grid cell (Figure 1D). The MODFLOW component of GSFLOW also relies on a user-specified stream network; stream segments represent tributaries, and the intersection of a stream segment with MODFLOW grid cells defines stream reaches (Figure 1A, D).

GSFLOW uses a daily computational time step for both the PRMS component and MODFLOW component. Flows are exchanged between each component at each time step. Multiple MODFLOW “stress periods” can be invoked to represent different subsurface boundary conditions within a simulation period, but their lengths must be integer days.

2.1.2 Process description

Table 1 in the GSFLOW manual (Markstrom et al., 2008) lists all PRMS modules, MODFLOW stress packages, and GSFLOW modules used by GSFLOW in the different modes. This section includes a brief description of the main processes they represent.

The PRMS component of GSFLOW includes various modules that can convert commonly available climate data into complete forcing inputs needed for model simulations. These modules include different methods for determining potential solar radiation, potential evapotranspiration, and snow accumulation / depletion; they also include different schemes for spatially distributing data from one or a few observations points over the entire watershed.

Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2017-321
Manuscript under review for journal Geosci. Model Dev.
Discussion started: 30 January 2018

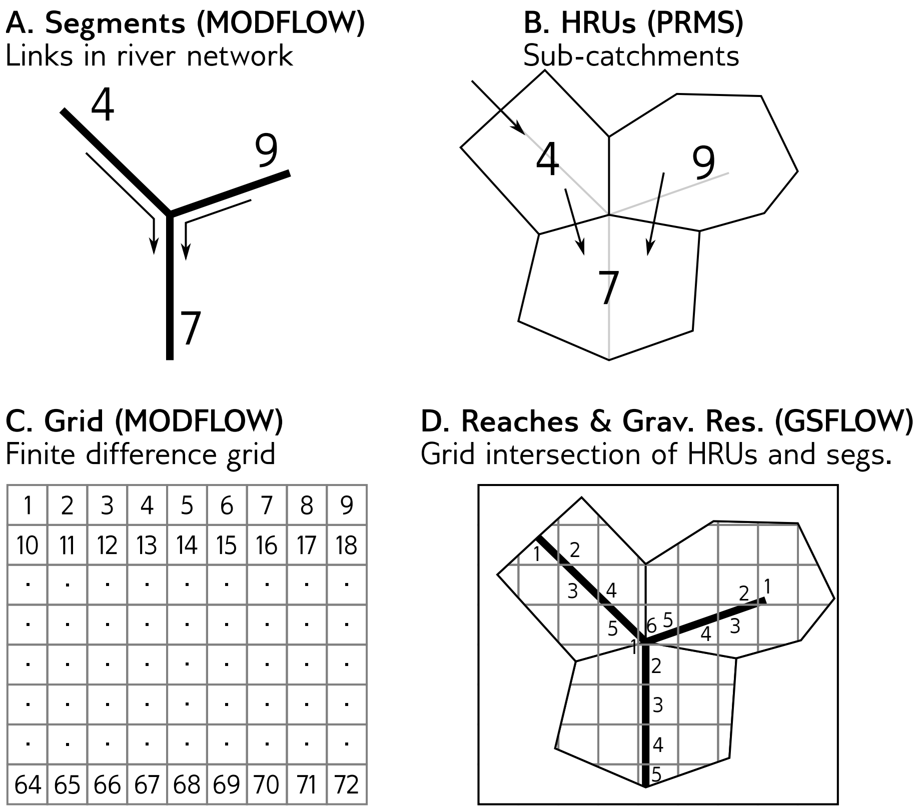

Figure 1. Major features of the GSFLOW geometry. **A.** Each segment is one link in the network. At each node, two tributary segments combine to flow into a single segment. Each is numbered, but need not be in any particular order. **B.** Flow in each of the sub-basin HRUs is routed directly to a corresponding stream segment. The arrow on the upper left indicates that flow from outside of the representative tributary junction may also be part of the drainage network. Our topological approach to defining HRUs allows HRUs to be numbered the same as the stream segments. Our code is written in such a way that future developments can relax this symmetry. **C.** MODFLOW operates on a grid that underlies the PRMS-based stream network and HRUs; each cell has a unique ID that is sequentially numbered. **D.** Gravity reservoirs are defined by the intersection of the PRMS HRUs and the MODFLOW grid. "Reaches" are defined as the section of each PRMS stream segment that lies within a single MODFLOW grid cell, and are numbered sequentially downstream as shown.

For unsaturated zone flow, PRMS does not implement Richards equation but instead applies efficient calculations to determine inputs and outputs for each HRU as well as exchanges among the three conceptual reservoir types within an HRU (GSFLOW manual Fig 19, Table 9). The "capillary zone" reservoir represents water held by capillary forces; it receives water through infiltration (based on parameter *pref_flow_den*) and loses water through evaporation and transpiration (based on parameters *soil_moist_max*, *soil_rechr_max*, and *soil_type*). After reaching field-capacity (parameter *soil_moist_max*), water transfers from the capillary zone to "gravity reservoirs," where water can flow horizontally as slow interflow (based on parameters *slowcoef_lin* and *slowcoef_sq*) and drain vertically into the deeper subsurface domain handled by MODFLOW (based on parameters *ssr2gw_rate*, *ssr2gw_exp*, and *ssrmax_coef*); gravity reservoirs can also receive groundwater discharge from the

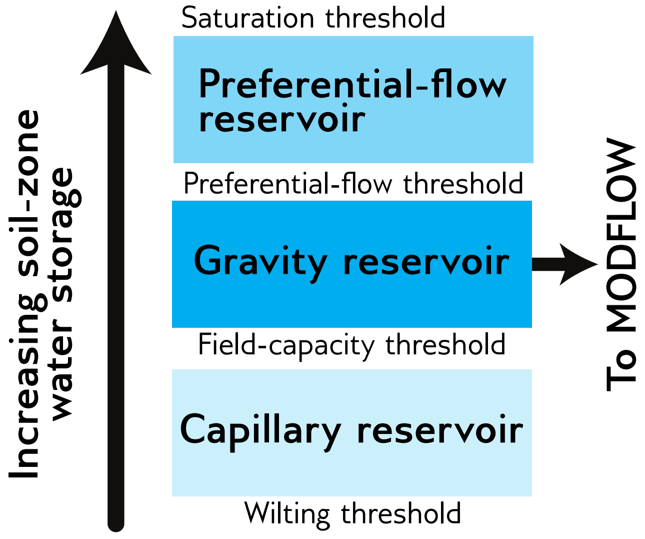

Figure 2. (Adapted from Markstrom et al. (2008), Figure 12) Soil water storage reservoirs in the PRMS component of GSFLOW. Within each HRU, soil water accounting calculations are carried out for three conceptual reservoirs in the order of increasing water storage and according to user-specified parameters. Climate forcing applies to the capillary reservoir, the gravity reservoir exchanges water with the deeper unsaturated and saturated zones represented by the MODFLOW component of GSFLOW, and Dunnian runoff and fast interflow occurs in the preferential-flow reservoir.

.

MODFLOW component when hydraulic head values exceed the lower limit of the soil zone. A fraction of gravity reservoir storage moves to the "preferential-flow reservoir" (based on parameters *pref_flow_den* and *sat_threshold*), where fast interflow occurs (based on parameters *fastcoef_lin* and *fastcoef_sq*). If the preferential-flow reservoir becomes full (based on parameter *sat_threshold*), then water exits the soil zone as Dunnian (saturation-excess) runoff. Hortonian (infiltration-excess) runoff calculations apply for impervious fractions of HRUs (set by parameter *hru_percent_imperv*). Surface runoff and interflow are routed between HRUs, using a cascading flow scheme that follows user-specified indexing of linked HRUs, and eventually reaches the stream network. A cascading HRU network is not needed for our domain formulation, which comprises sub-basin HRUs that route water directly to stream segments (Section 3.2) .

The MODFLOW component of GSFLOW computes water flow in the deeper unsaturated zone (UZF stress package), streams (SFR package), and saturated groundwater units (BCF, LPF, or UPW flow packages). Unsaturated zone flow is calculated using a kinematic-wave approach, which assumes that capillary (pressure gradient) flow is negligible compared to gravity-driven flow. Capillary-dominated effects are instead represented in the soil zone of the PRMS component described above. Unsaturated zone flow in the MODFLOW component is calculated as waves representing wetting and drying fronts. Gravity reservoir drainage from the PRMS component flows to the top of the unsaturated zone of the MODFLOW component, unless the water table is above the soil-zone base – defined by the top of the MODFLOW domain – in which case the gravity reservoirs drain directly to the saturated zone. Saturated zone simulations (MODFLOW) employ finite difference solutions to the groundwater flow equation.

Streamflow, as calculated by the MODFLOW component, includes inputs from upstream reaches, surface runoff and interflow from the PRMS component, base flow from the saturated zone discharge, and flows from possible underlying unsaturated areas. Outputs include flow to downstream reaches, leakage to groundwater, and flows to possible underlying unsaturated areas. Discharge across the streambed follows Darcy calculations with specified streambed hydraulic properties. User-specified additional inputs and outputs are also allowed. Five different options exist for stream discharge and head computations (parameter *ICALC*). The user can specify stream depths for each reach; apply Manning's equation to an assumed wide rectangular channel; apply Manning's equation for an eight-point-based channel and floodplain geometry; apply at-a-station power-law relationships between discharge, flow width, and flow depth (Leopold and Maddock, 1953); or specify an input look-up table of hydraulic geometries for each segment. Streamflow can be simulated as either steady-state flow (parameter *IRTFLG* = 0), where outflow to the next stream reach balances inputs, or as transient flow (parameter *IRTFLG* > 0), using a kinematic wave formulation for surface water routing in channels, which applies the assumption that the water surface slope approximates the friction slope, and therefore negates backwater effects.

It should be noted that some modifications were made to the original stand-alone PRMS and MODFLOW codes for their use in GSFLOW. Notably, the soil-zone structure of PRMS is significantly altered to facilitate linking to a MODFLOW subsurface domain. Other modifications are noted in the GSFLOW manual (see sections on "Changes to PRMS" and "Changes to MODFLOW-2005"). An additional feature starting in version 1.2.0 that is not described in the original manual is the inclusion of MODFLOW-NWT (Niswonger et al., 2011), a more numerically robust update to MODFLOW-2005 for groundwater flow.

2.2 GRASS GIS

GRASS GIS is an open-source, multi-purpose, and cross-platform geographic information system (Neteler and Mitasova, 2008; Neteler et al., 2008, 2012) that supports utilities for efficient raster and vector computations (Shapiro and Westervelt, 1994; Mitasova et al., 1995; Ŝúri and Hofierka, 2004; Hofierka et al., 2009). It includes both graphical and command-line interfaces, and may be driven by shell or Python scripts. It supports both 2D and 3D raster and vector data and includes SQL-based attribute table management. GSFLOW-GRASS utilities are written for the current most recent stable release version of GRASS GIS, v7.4. This supports Python scripting for both high-level built-in commands and for low-level access to database entries and vector geometries (Zambelli et al., 2013). We take advantage of these capabilities to develop an automated workflow to build GSFLOW inputs through GRASS GIS.

We chose GRASS GIS as the interface to develop inputs because (1) it is open-source and cross-platform; (2) it enforces rigid vector topology, and it is critical for building stream networks; (3) its generic Python scripting library and PyGRASS Application Programming Interface (API) make it easy to develop new extensions; (4) these extensions may be added to the official subversion (svn) repository, from which they can be automatically downloaded and installed on users' computers using the **g.extension** command, and (5) it supplies a GUI and command-line interface (CLI) that are consistent with one another. These GUI and CLI interfaces are not required for GSFLOW-GRASS, because the GRASS GIS component is handled mostly behind-the-scenes by a batch processing Python script (**buildDomainGRASS.py**); however, they allow end-users to re-run certain portions of the process and/or produce their own workflows using the GSFLOW-GRASS extensions as building blocks. The open-source aspect is motivated by the need for water assessment and planning tools in the developing world (Pal et al., 2007), and these combined with the interchangeable and consistent GUI and CLI help users to generate their own advanced customizations of GSFLOW-GRASS.

3 Methods

GSFLOW-GRASS strikes a balance between generating a ready-to-go GSFLOW implementation and providing flexibility to customize applications. Out-of-the-box, and with only a few steps to set up the model with the user's computer directory system, the toolkit can be applied to any DEM to readily produce GSFLOW model simulations for a watershed. Then, for further tuning, all scripts in the toolbox are open-source and commented to allow changes to any parameter and development of optional GSFLOW capabilities not included in the default GSFLOW-GRASS implementation. While many popular hydrologic model implementation programs have GUIs (e.g., ModelMuse (Winston, 2009), Visual Modflow (Waterloo Hydrogeologic Inc., 2011), Hydrus (Simunek et al., 2009), ArcSWAT (Neitsch et al., 2002), MIKE-SHE (Butts and Graham, 2005)), which are easiest for novice model users, GUIs can be challenging to develop for cross-platform implementations, are often unstable, and generally support less flexibility for customization. Thus, we chose a mostly command-line GRASS-GIS / Python program-based approach. GSFLOW-GRASS has been designed and tested for use on Linux and Windows operating systems.

3.1 User-specified Settings and Model Inputs

For a seamless execution of GSFLOW-GRASS, a Settings text file is used for specifying all inputs needed, including those for the GRASS GIS domain builder, GSFLOW input-file builder, and visualization components described below. We note, however, that only certain model inputs may be set through the Settings file. GSFLOW requires a daunting number of different model inputs (nearly 200 parameters for the PRMS sub-component alone). For ease-of-use, this toolkit applies default values for most inputs, and only includes a handful of application-specific and commonly adjusted inputs in the Settings text file. When using the default implementation, the user only modifies the Settings text file and does not need to directly handle any GRASS GIS tools or Python scripts. However, for flexibility, additional parameters may be changed in the open-source scripts by searching for the parameter names specified in the GSFLOW manual, including those mentioned in Section 2.1.2 . All inputs from the Settings file are read in and processed by the **ReadSettings.py** script.

The first section of the Settings file is "paths"; this is where computer-specific names and directory locations are listed, such as the GSFLOW executable file and the directory in which simulations will be run. The **ReadSettings.py** script creates a directory structure to organize all GIS and simulations files. This imposed directory structure supports easy exchange between the different toolkit modules and allows the use of relative directory names, which facilitates the sharing of model files across computers systems and between users.

The “GRASS” section contains both geospatial inputs and other entries used by the GIS portion of GSFLOW-GRASS. The former includes the name of the user-provided DEM file, spatial resolution parameters needed to build the computational domain, and the approximate catchment outlet point. The latter includes a number for the GSFLOW input *ICALC*, which encodes for the hydraulic geometry and flow resistance relationship based on flow width and surface roughness.

The “run_mode” section allows the user to execute GSFLOW in either “spin-up” or “restart” mode (Regan et al., 2015). Spin-up simulations start with a preliminary MODFLOW steady-state execution using specified infiltration to the MODFLOW domain (see below) to calculate reasonable conditions for initializing a subsequent transient simulation within the spin-up run. The steady-state step can be essential for obtaining initial conditions for the groundwater system for numerically convergent MODFLOW results and helps with reaching more realistic transient state solutions. At the end of a spin-up run, final PRMS and MODFLOW state variables are saved in files that can be specified in the run_mode section to initiate “restart” runs without the preliminary steady-state period.

The “domain” section is used to specify the temporal window and the vertical discretization, which is not generated by the GRASS module. GSFLOW-GRASS allows for multiple vertical layers, each of which can have a different horizontally uniform thickness, with the top and bottom of each layer mirroring land surface topography. This section is also used to specify the start date and end date of the simulation.

The “custom_inputs” section is used to set three key model inputs: climate forcing, saturated hydraulic conductivity, and infiltration rate (infiltration is used for the preliminary steady-state MODFLOW calculation in spin-up runs). Although the PRMS component of GSFLOW includes various modules to spatially distribute climate variables, the Settings file only takes inputs for the “climate_hru” option (see Section 3.3.1). Both hydraulic conductivity and infiltration can be set to either a single value to be applied uniformly over the entire model domain, or to spatially distributed values stored in separate files. There is an option for invoking a script from GSFLOW-GRASS that generates spatially discretized hydraulic conductivity. Based on our example tests in Section 4, adjusting hydraulic conductivity and infiltration is important for generating numerically convergent results and establishing reasonable initial conditions for transient simulations. Modules in the toolkit by Gardner et al. (2017) generates climate forcing, soil type, and land cover inputs that can make use of national databases; for GSFLOW-GRASS, users provide their own inputs. The USGS’s GIS Weasel tool (Viger and Leavesley, 2007) may be used for deriving PRMS parameters from physical data sets such as STATSGO.

3.2 GRASS GIS: Topography-derived model inputs

We built ten new GRASS GIS “extensions”, also called “add-ons”, which are commands that work alongside existing GRASS GIS functionality to transform user specifications and a single digital elevation model (DEM) into a set of GSFLOW inputs. These inputs are stored as raster data sets and SQL database tables attached to vector geometries, and then exported to ASCII files that are later parsed by the Python input-file builders scripts (Section 3.3). The separate ASCII files allow users to set up the spatial structure of the model once using GRASS GIS, and then perform multiple runs for parameter estimation or scenario tests without having to repeat the domain construction.

The toolkit's GRASS GIS extensions define HRUs topologically – that is, based on the drainage network. This follows both the natural discretization of the landscape and the architecture of PRMS, which was developed to support flow routing through sub-catchments (Markstrom et al., 2015). This approach is complementary to the grid-cell HRU approach of (Gardner et al., 2017).

In the first step of the GRASS workflow, GRASS GIS imports a user-provided DEM, removes "off-map" cells that may have flow contributions from outside of the DEM, hydrologically corrects the DEM elevations, and builds a drainage network (Metz et al., 2011). We then construct a Hortonion drainage network and the sub-catchments associated with each segment in that network using the **r.stream.*** set of extensions by Jasiewicz and Metz (2011). In creating this network, the user must supply a threshold drainage area for a stream segment to be defined; this is done with the help of our **r.cell.area** command, which automatically creates a raster of grid cell sizes. Smaller drainage area thresholds generate a larger set of HRUs, and the largest number of HRUs that can be defined is a function of the total drainage basin area and the resolution of the DEM.

The next step is to map the connections between each segment in the tributary network. To do this, we developed an extension called **v.stream.network**. Each stream segment has a unique positive integer identifier, called a "category" in GRASS GIS. For each segment in the drainage network, v.stream.network writes the category value of the immediately downstream stream segment to the "tostream" column in its associated attribute table row. Any stream segment exiting the map area is given a "tostream" value of 0. After this, we limit the study area to a single drainage basin using our **v.stream.inbasin** extension. At this point, the drainage network geometry and topology is developed, and we move on to primarily using our new **[r/v].gsflow.*** set of extensions to populate attribute values and link the topographically driven surface-water flow geometry with a regular subsurface grid for MODFLOW.

The final step within the GRASS workflow uses a sequence of new commands, designated by "gsflow," and associated built-in GRASS GIS utilities to provide the geometry, topology, and much of the stream parameters needed to build GSFLOW input files (Section 3.3). **v.gsflow.segments** numbers each river channel segment (Figure 1A) and sets the attributes that define its hydraulic geometry, channel roughness, input discharge beyond that from precipitation for the upstream-most stream segments, input diffuse runoff, and direct precipitation on the stream. In its current form, **v.gsflow.segments** applies Manning's equation to a wide channel (*ICALC*$=1$) and allows the user to set a single channel width and Manning's n (in-channel roughness coefficient for flow resistance) across the whole domain. This is designed to be a starting point for simulations that is not necessarily realistic; field data on channel geometries come in a variety of forms, which would be difficult to accommodate in a generalized tool. Instead, we encourage users to modify the GRASS GIS database tables. This can be performed via **v.db.update** within GRASS GIS; by modifying the GRASS GIS extensions that we have built to enable additional user-set options that specify hydraulic geometry; and/or to edit the exported files, **segments_4B_UPSTREAM.txt** and **segments_4C_DOWNSTREAM.txt** in the **GIS** subfolder. **v.gsflow.hruparams** creates and populates attributes of each HRU (Figure 1B). **v.gsflow.grid** aligns and builds the MODFLOW grid along the higher-resolution (i.e. original flow-routing) DEM cell edges (Figure 1C); it also identifies the grid cell directly downstream of the drainage outlet, which can be used to generate a constant-head boundary condition for improved simulation performance (see Section 3.3.3). **r.gsflow.hydrodem** generates a hydrologically corrected DEM at the resolution of the MODFLOW grid, which the user can make coarser than the flow-routing DEM for computational efficiency.

In order to maintain downstream flow while downsampling, the lowest-elevation pixel that corresponds to each MODFLOW stream cell is selected as its elevation, corresponding to the stream channel within these cells, while non-stream MODFLOW cells are assigned the mean elevation from the higher-resolution DEM. **v.gsflow.reaches** and **v.gsflow.gravres** construct the "reaches" and "gravity reservoirs", which are the intersection of segments and HRUs, respectively, with each MODFLOW grid cell (Figure 1D). The reaches include values for the thickness of the stream bed sediment (defaults to 1 meter) and its hydraulic conductivity (defaults to 5 m/d, characteristic of sand and gravel). Default values are set by **buildDomainGRASS.py**, though the user can modify these. **buildDomainGRASS.py** then exports the rasters, which comprise a "basin mask" and the hydrologically corrected DEM, both at the resolution of the MODFLOW cells, using the built-in commands **g.region** and **r.out.ascii**. After this, **v.gsflow.output** creates comma-separated variables (CSV) files for the segments, HRUs, reaches, gravity reservoirs, the pour point at the outlet of the basin, and constant-head boundary condition cells. Finally, **buildDomainGRASS.py** exports the derived vector maps in shapefile format using the built-in GRASS GIS **v.out.ogr** command; these include the HRUs, gravity reservoirs, MODFLOW grid, full study basin area, stream segments, stream reaches, pour point, and downstream boundary-condition cells. All of these outputs are written to the **GIS** subdirectory of the project folder, whose location is specified in the Settings file. GSFLOW supports more input options than we have defined for our GRASS GIS **v.gsflow.*** commands, but we have included the most common options, and the structure of the open-source code is straightforward to modify to include additional attribute values.

User customization of the GRASS GIS component of GSFLOW-GRASS is supported by the modular architecture of the GRASS GIS extensions. Users who are comfortable with writing Python code may add additional extensions and/or add or modify options to existing extensions. This may also require additional options to be added to the settings file and to the settings-file reader, **readSettings.py**. Users can then develop their own version(s) of **buildDomainGRASS.py** that incorporate these changes.

GSFLOW-GRASS aligns each stream segment with a sub-basin HRU (Figures 1A 1B). This simplifies the topology, but requires that each HRU has only a single slope, aspect, set of vegetation, and surface hydrologic parameters. For finer-resolution variability, the user can decrease the threshold drainage area to define a channel, and link these to other (e.g., land-surface) properties. This connection must be built outside of GSFLOW-GRASS's pre-made workflow, but would be possible through additional open-source GRASS GIS extensions that build off of the present work. Furthermore, although HRUs and stream segments perfectly overlap as part of the same network with GSFLOW-GRASS, each has its own distinct set of database entries, enabling the structure to be modified to represent finer-resolution heterogeneity between the channel and the surrounding hillslopes.

The domain-building procedure described above is automated through the **buildDomainGRASS.py** script. The user executes this script within a GRASS GIS location with a user-specified map projection. This script takes inputs from the Settings text file, implements the domain-building workflow, and produces ASCII files used by GSFLOW-GRASS's Python input-builder scripts. This DEM-based approach relies on minimal and easily available data to set up a GSFLOW computational domain.

Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2017-321
Manuscript under review for journal Geosci. Model Dev.
Discussion started: 30 January 2018

Geoscientific Model Development Discussions Open Access EGU

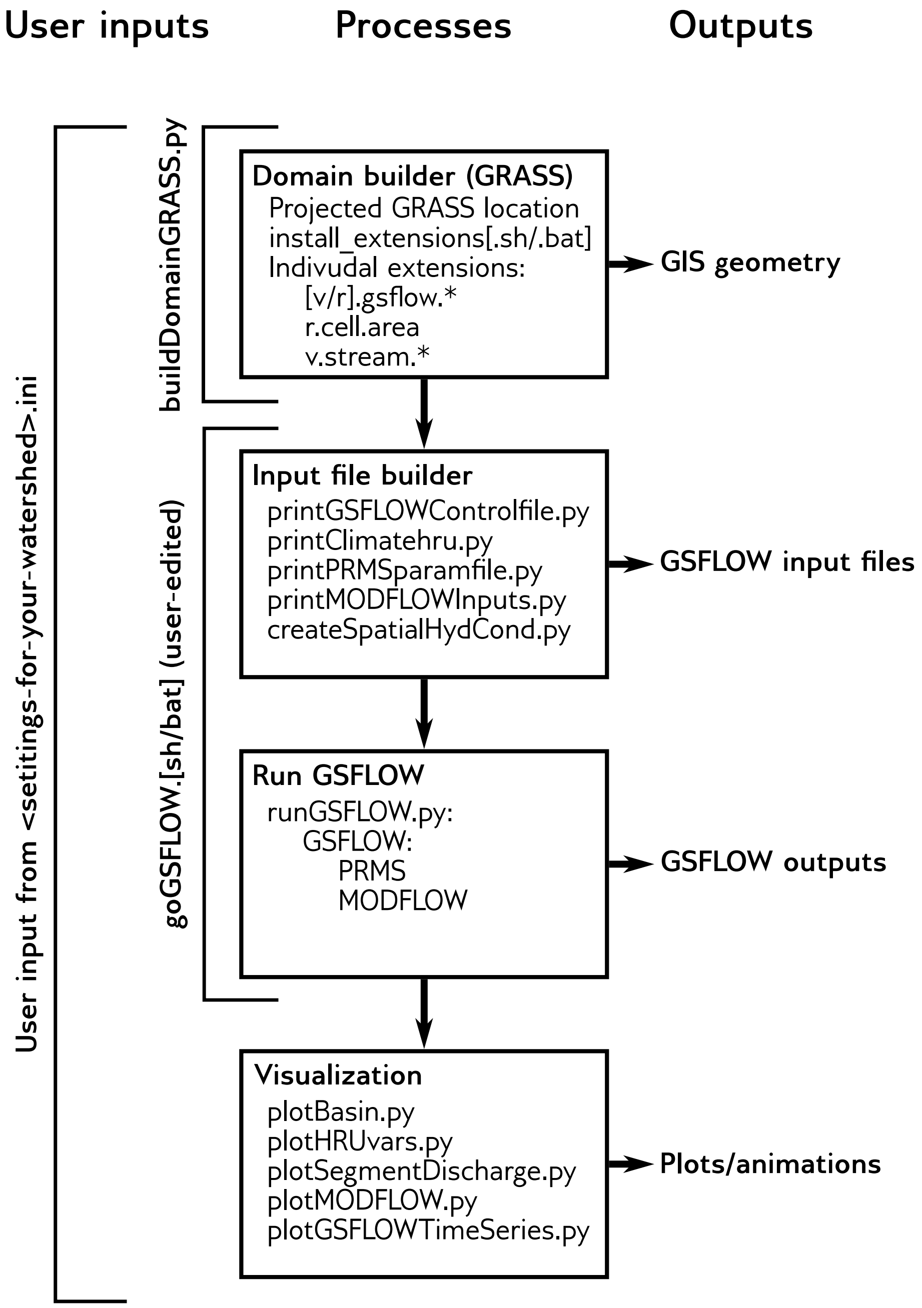

Figure 3. GSFLOW-GRASS workflow. The user: (1) creates a ***.ini** file based on their study catchment; (2) creates a projected GRASS GIS location; (3) runs **buildDomainGRASS.py**; (4) edits and runs **goGSFLOW.py**. After this, they may use GSFLOW-GRASS's visualization tools to study the GIS and model outputs.

.

3.3 GSFLOW Input File Builder

With the model domain constructed using the GRASS GIS workflow, GSFLOW input files can be created with GSFLOW-GRASS's builder scripts for the GSFLOW control file, PRMS-type input files, and MODFLOW-type input files. Most of the new features unique to GSFLOW (not in stand-alone PRMS or MODFLOW) follow the same Modular Modeling System input-data file format (Leavesley et al., 1996) as PRMS, which includes the use of a "control file" as the main interface file, the use of "modules" for different computational options, and the PRMS input file syntax. In contrast, MODFLOW uses a "name file" as its main interface file, implements "packages" for computational options, and follows its own file syntax. Execution of the following builder scripts is automated through the toolkit's Run file (Section 3.4), but the builder scripts may be customized for extensions beyond the default implementation.

3.3.1 GSFLOW Control File

The GSFLOW control file is the highest level input file and is created by the **printGSFLOWControlfile.py** script in the GSFLOW-GRASS toolkit. The toolkit is streamlined for configuring the integrated mode of GSFLOW (set through the "model_mode" parameter).

Inputs for the control file parameters are organized under six numbered sections in **printGSFLOWControlfile.py**. The script sets parameters related to climate forcing, time domain, and run mode based on what the user specifies in the Settings file; all other parameters are pre-set to default values. Further customization of control file parameters (stored in the list variable *con_par_name*) requires simply changing default values (in the corresponding list variable *con_par_values*) in the script. The first two sections are non-optional and include details about the simulation execution and module choices. The third section establishes spin-up versus restart run modes based on Settings file entires. Section 4 contains a customizable list of HRU-distributed output variables to be printed; visualization scripts are available in GSFLOW-GRASS to create movies for these variables (Section 3.5). Section 5 also includes a list of output variables, but for whole basin-level values; the default gsflow output file will already include the major basin-level variables, which makes this section unnecessary for most cases. The last optional section is for running the model in a debugging mode.

Note that the default implementation of this toolkit uses the "climate_hru" module for precipitation and minimum and maximum daily temperature; this means that the model will employ pre-existing files containing data already specified by HRU. The PRMS component of GSFLOW does include other modules for distributing data from one or a handful of weather stations, but these typically require application-specific empirical parameters that are difficult to incorporate in a generic toolkit. Use of the the "climate_hru" module provides flexibility for the user to implement their own spatial interpolation or extrapolation methods. However, for a quick set-up of GSFLOW, we do provide a script for uniformly applying a single climate data series over all HRUs to create climate_hru files (more details in Section 3.3.2). GSFLOW-GRASS's default implementation also uses the Priestley-Taylor formulation for potential evapotranspiration calculations (Markstrom et al., 2008). This module was chosen because of its reliance on only air temperature and solar radiation (calculated by the PRMS component of GSFLOW), and because of the relative ease of accounting for different vegetation properties through the parameter *pt_alpha* (in the PRMS parameter file, Section 3.3.2).

After the six parameter input sections in **printGSFLOWControlfile.py**, the script builds the control file and then generates an executable file (shell script for Linux and batch file for Windows) for running GSFLOW with the control file. After all other input files are created, this executable gets called by the toolkit's automated Run file (Section 3.4). The executable can also be used to run GSFLOW outside of the GSFLOW-GRASS toolkit.

3.3.2 PRMS-type Input Files

Input files required for the PRMS component of GSFLOW are the parameter file ("param_file" in the control file), which includes empirical surface and soil zone properties, and the data file ("data_file" in the control file), which includes climate observations for the spatial interpolation/extrapolation algorithms. If the "climate_hru" module is selected, as it is in the toolkit's default implementation (Section 3.3.1), then individual input files with HRU-distributed climate variables must also be specified. GSFLOW-GRASS includes the script **printClimatehru.py**, which takes daily observations from a single file and distributes them uniformly over all HRUs. The toolkit handles the minimum required climate variables – daily precipitation, maximum temperature, and minimum temperature, and it is set up to be readily extended to also include humidity, solar radiation, and/or wind speed. This spatially-uniform approach is appropriate where the size of a rainstorm is significantly greater than the size of a catchment and climatic variables vary only weakly with slope and aspect; larger and higher-relief catchments require spatially-distributed climate inputs for realistic outputs, and these require custom inputs from the end user.

The parameter file is created by the script **printPRMSparamfile.py**. The script first includes a section for domain dimensions, including the list variable *dim_name* for the different dimension names and *dim_value* for their corresponding values, and then a section for parameters, including the list variable *par_name* for the different parameter names and *par_value* for their corresponding values. Both sections include various entries that take values parsed from the GRASS GIS domain builder outputs, as indicated in the comments in the script. Because of PRMS's conceptual soil moisture regimes, it requires a substantial number of parameters inputs related to the soil and vegetation that cannot easily be specified without calibration. As a default to help the user get GSFLOW up and running, most parameter values in **printPRMSparamfile.py** are preset, mostly using calibrated values from the Sagehen watershed example that was distributed with the GSFLOW model version 1.2.1. We have indicated with the comment "# *** CHANGE FOR SPECIFIC SITE" those parameters that could also be altered based on known characteristics of one's watershed site. This includes various soil and land-cover inputs, such as *soil_type* (sand, loam, or clay), *cov_type* (bare soil, grasses, shrubs, or trees), *transp_end* (end month of transpiration, for phenology), and *pt_alpha* (Priestley-Taylor parameter α, which can be based on literature values). In addition to these highlighted parameters, users can review all parameters to determine whether others could be particularly important for their specific application. These may include some of the parameters mentioned in Section 2.1.2 that determine exchanges between different soil-zone reservoirs. Rigorous calibration of PRMS parameters can eventually be carried out with inverse codes such as PEST (Doherty, 1994) or UCODE (Poeter and Hill, 1998, 1999).

3.3.3 MODFLOW-type Input Files

GSFLOW requires input files for each MODFLOW package utilized, which can include any of the packages listed in Table 1 of the GSFLOW manual (Markstrom et al., 2008, Appendix 1, p. 176-226 provides details). Our toolkit by default creates a relatively general MODFLOW set-up, which includes required input files and omits most optional ones, such as the Well package. Our Python library **MODFLOWLib.py** consists of functions for creating: four Basic package input files (name file, basic package file, discretization file, and the optional output control file for customizing output files), two different groundwater flow package options (the Layer-Property Flow (LPF) from MODFLOW-2005 and the Upstream Weighting Package (UPW) from MODFLOW-NWT), the numerical solver package (Preconditioned Conjugate Gradient (PCG) for LPF and Newtonian (NWT) input file for UPW), the Streamflow-Routing package (SFR), and the Unsaturated-Zone Flow package (UZF).

The script **printMODFLOWInputs.py** calls the functions from **MODFLOWLib.py** to create a set of internally consistent input files that incorporate the domains constructed by the GRASS-GIS workflow (Section 3.2) and conform to the simulation directory structure established through our **ReadSettings.py** utility. By default, **printMODFLOWInputs.py** calls the MODFLOW-NWT UPW/NWT flow package instead of the MODFLOW-2005, because of the superior numerical performance of the former in tests with steep elevation gradients (e.g., Section 4.1). If desired, users can easily switch to the LPF/PCG formulation from MODFLOW-2005 by setting *sw_2005_NWT* = 1 in **printMODFLOWInputs.py**.

Input files created outside of our toolkit for a stand-alone MODFLOW model implementation of identical discretization will for the most part be usable with the integrated GSFLOW model. However, as indicated in Table 1 of the GSFLOW manual, some MODFLOW packages were modified for their use in GSFLOW. Advantages of implementing our toolkit over using pre-created MODFLOW input files are that it already incorporates these GSFLOW modifications, it automatically uses the GRASS-GIS builder results for the domain, and it guarantees consistent directory structure with the rest of the input files and the visualization scripts.

The GSFLOW-GRASS toolkit also offers an optional script **createSpatialHydCond.py** for generating spatially distributed hydraulic conductivity fields for the upper layer based on elevation and/or distance from the stream network, with the assumption that lower elevations and/or riparian corridors have higher hydraulic conductivity properties. Because application-specific entries cannot easily be generalized for input through the Settings file, users should directly customize elevation and stream distance thresholds, as well as corresponding hydraulic conductivity values, at the top of the **createSpatialHydCond.py** script. This script will automatically import domain information from the Settings file and export results to the file location specified by the Settings file. **createSpatialHydCond.py** serves as a ready-to-go tool for creating physically plausible hydraulic conductivity patterns, and it provides an example for how users can create their own scripts to customize spatially distributed inputs. A similar type of script could create spatially distributed infiltration fields for the preliminary MODFLOW steady-state simulation in spin-up runs (see discussion of *finf* entry in the Settings file in Section 3.1). These tools can provide preliminary inputs to jump-start GSFLOW model implementations. However, realistic construction of hydrogeologic frameworks relies on data from sources such as well logs, geologic maps, geophysical measurements, and pumping tests (Reilly, 2001; Reilly and Harbaugh, 2004). Properties for stream segments and reaches – such as streambed hydraulic conductivity, and unsaturated hydraulic properties below the streambed – are set to default values that can be changed through the GRASS GIS component. By default, the streamflow calculation is set to use Manning's equation assuming a wide rectangular channel ($ICALC = 1$). These may be modified by adjusting the interface to the GRASS GIS modules (see Section 3.2 and the accompanying user's guide).

3.4 GSFLOW Run File

For the user's convenience, the GSFLOW-GRASS toolkit includes an executable Run file (a shell script for Linux: **goGSFLOW.sh**, and a batch file for Windows: **goGSFLOW.bat**) that uses a specified Settings file and runs all of the above input-file builder scripts as well as the script **runGSFLOW.py**, which launches the GSFLOW simulation. Thus, as long as the user does not wish to change the default implementation beyond features in the Settings file, no direct interface with the Python program or code is required to run GSFLOW-GRASS. This allows for "quick-start" implementations of GSFLOW, which can substantially lower the barrier to entry for using this model.

The Run file can be implemented after the model domain is generated through **buildDomainGRASS.py**. The toolkit separates out the GRASS domain builder module from the Run file, because users will typically only need to construct their domain once, but will want to re-try simulations with various parameter inputs.

After preliminary quick-start simulation tests, users can further customize their runs by taking advantage of the modular structure of the toolkit, which has a separate script for each input file. For example, to target specific aspects of the model, such as the surface runoff properties, corresponding parameters may be adjusted in the PRMS parameter file by editing and re-running **printPRMSparamfile.py**. Select input-file builder scripts can be run either within Python, or by editing the executable Run file.

3.5 Visualization Tools

Our toolkit includes post-processing Python scripts that employ the Matplotlib plotting library (Hunter, 2007) for visualizing the domain discretization, key MODFLOW inputs, and model output results. The model discretization for the PRMS component of GSFLOW is exported from GRASS GIS as a set of standard vector GIS files (shapefiles). Our Python plotting scripts use these shapefiles to create figures of the surface HRU and stream segment discretization (**plotBasin.py**), and to generate movies of HRU-distributed and stream segment-distributed variables (**plotHRUvars.py** and **plotSegmentDischarge.py**). These output variables (e.g., evapotranpsiration and streamflow) are set through *aniOutVar_names* in the GSFLOW control file (see Section 3.3.1). The exported shapefiles may also be used to visualize model results with standard GIS packages (e.g., QGIS) outside of GSFLOW-GRASS.

For the MODFLOW component of GSFLOW, the toolkit's script **plotMODFLOW.py** plots spatially distributed layer elevations, hydraulic conductivity, and a map of active computational grid cells. The script also plots spatially distributed MODFLOW simulations results over time, including for hydraulic head, change in head, water table depth, and recharge from the unsaturated zone. For storage efficiency, the toolkit creates and reads in head and unsaturated zone output files in binary format.

For basin-total GSFLOW results, the Python script **plotGSFLOWTimeSeries.py** generates time series lines for user-selected variables from the main GSFLOW csv output file. Names of all variables, along with their descriptions and units, are listed in **GSFLOWcsvTable.py**, which is imported into **plotGSFLOWTimeSeries.py** to ensure consistency in figure labels and axes.

The visualization scripts can be run using a command-line parser and/or by editing plot options that appear near the top of each script. More advanced users may modify the body of the scripts to change to features such as axis intervals or color schemes. For users who want to adjust the scripts, we suggest running them in the iPython interactive programming console (Pérez and Granger, 2007), which is also incorporated into the Spyder integrated development environment (IDE). Although this visualization approach requires some familiarity with Python and/or command-line argument parsing, it accommodates the wide range of plotting preferences. All plots and videos may be displayed as on-screen figures (in raster or vector formats, using the interactive Matplotlib window), and may be saved as images (interactively) or videos (***.mp4** format) as defined by inputs to the plotting script.

Other existing no-fee USGS GUI programs for MODFLOW also provide visualization capabilities, and using these with the input and output files produced with GSFLOW-GRASS may be possible. In particular, GW Chart (Winston, 2000) can be directly implemented for plotting basin-level time series results. Additionally, Model Viewer (Hsieh and Winston, 2002) and ModelMuse (Winston, 2009) are able to read in and plot spatially variable head results from binary files with the extension ".bhd," but this does require manual post-processing steps. For Model Viewer, the user needs to copy all MODFLOW input and output files to a new folder inside the Model Viewer project directory and select the namefile when prompted. For Model Muse, the user must first delete the line that starts with "IWRT" from the name file in order to load the project into the program. Once the project settings are loaded into ModelMuse, the user can use the "import model results" tool to select the binary head file.

4 Examples

As examples of the application of GSFLOW-GRASS, we have chosen the water-stressed Shullcas River Watershed, Junín Region, Peru, which is experiencing rapid glacier retreat; Water Canyon on Santa Rosa Island off the coast of California, which has undergone land-cover change impacts; and the formerly-glaciated Cannon River, which flows from intensely farmed uplands into an incised bedrock valley in Minnesota, USA (Figure 4). All regions contain complex hydrology with interactions between surface water and groundwater and are exemplars of practical management concerns. Together they span a range of environments: high to low elevations, steep to low-gradient catchments, coastal to inland settings, tectonically-active to cratonal, and glacier-influenced to temperate. They are impacted by modern climate and land-use change impacts on glaciers and agricultural (water and soil) resources (Shullcas) (Gómez et al., 2014; Arroyo Aliaga et al., 2015; Travezan Adauto, 2015), agricultural runoff and fertilizers (Cannon River) (Kreiling and Houser, 2016), and grazing-induced erosion (Santa Rosa) (Minnich, 1982). Our choice of an example in the Peruvian Andes demonstrates how our entirely open-source modeling system may be applied to problems in the developing world.

Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2017-321
Manuscript under review for journal Geosci. Model Dev.
Discussion started: 30 January 2018
© Author(s) 2018. CC BY 4.0 License.

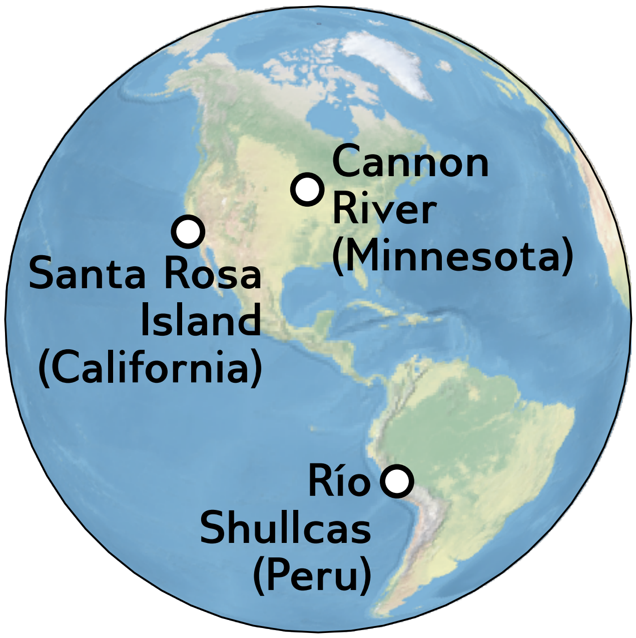

Figure 4. Our test sites span include the high Andes, a mountainous island, and a formerly-glaciated Mississippi tributary.

The three applications show that the GSFLOW-GRASS toolkit can readily generate GSFLOW inputs that produce numerically convergent simulations in a variety of topographies and hydroclimatic conditions. All three of these model exercises are purely schematic: no efforts were made to calibrate GSFLOW to field conditions. Figures 5–7 display sample inputs and outputs of the model simulations using the GSFLOW-GRASS toolkit for the three test sites.

4.1 Shullcas River Watershed, Peru

The first test case is from the Shullcas River Watershed, located in the central Peruvian Andes. Precipitation is highly seasonal, and water shortages are common during the dry season from May to September. The Huaytapallana Glaciers, which supply meltwater to the Shullcas River, are rapidly retreating (López-Moreno et al., 2014), causing concern over future water resources. However, a large proportion of the dry season stream discharge is composed of groundwater, driving the need to better understand groundwater-surface water interactions in the catchment. The steep topography and seasonal precipitation make the Shullcas River Watershed an apt testbed for examining the ability of GSFLOW–GRASS to represent surface-water–groundwater links in challenging terrain.

GSFLOW-GRASS was used to prepare a simple hydrological model based on the Shullcas watershed. The modeled portion of the watershed has an area of 170 km^2 and ranges in elevation from 3600 to 5500 m above sea level (a.s.l.). Using the GRASS domain-builder, the watershed was divided into 59 sub-catchment HRUs based on an ASTER 30 m resolution DEM (Tachikawa et al., 2011). The subsurface was represented by a single 200 m thick MODFLOW layer, with a horizontal discretization of 46 rows, each with a length 485 m, by 33 columns, each with a width of 492 m. After an initial steady-state stress

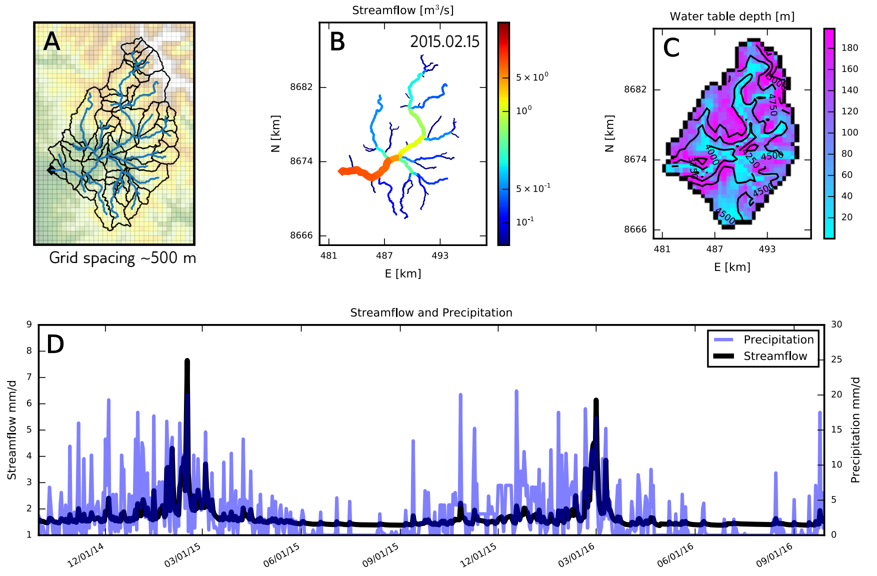

Figure 5. Río Shullcas, Peru. **(A)** Map with MODFLOW grid, HRU outlines, stream segments (blue), and digital elevation model. **(B)** Streamflow accumulates through this mountainous drainage network. **(C)** Groundwater is shallowest in low and flat areas. **(D)** In our simulations, peak discharge occurs late in the wet season, after significant antecedent moisture has built up within the catchment; essentially constant baseflow supports low but reliable discharge throughout the dry season.

period in the "spin-up" run mode, the simulation ran for approximately three years from August 26, 2013 to September 29, 2016. Meteorological data – including maximum temperature, minimum temperature, and precipitation – were taken from the Peruvian Meteorological Office (SENMAHI) online database, and are used to drive the model. Our Shullcas-based simulation does not represent glacier melt, but spatiotemporal results in Figure 5 show that GSFLOW can be useful for evaluating the potential for groundwater to buffer surface water resources in mountainous watersheds with high seasonal variability. For all examples, results are shown after the effects of the preliminary steady-state conditions become minimal.

4.2 Santa Rosa Island, California, USA

Santa Rosa Island is one of the Channel Islands of California, USA, and is part of the Channel Islands National Park. The island has an area of approximately 214 km^2 and is characterized by mountainous topography, with its highest point at 484 m.a.s.l. (Clark et al., 1990).

Hydrologic modeling of Santa Rosa Island has previously been performed by Jazwa et al. (2016), who applied the PIHM hydrologic model (Qu and Duffy, 2007) to the island in order to understand the relationship between prehistoric human settlement patterns and surface water availability. They reported streamflow characteristics for the 19 major drainages around the island during simulated hypothetical climate regimes that are wet, dry, and of average wetness.

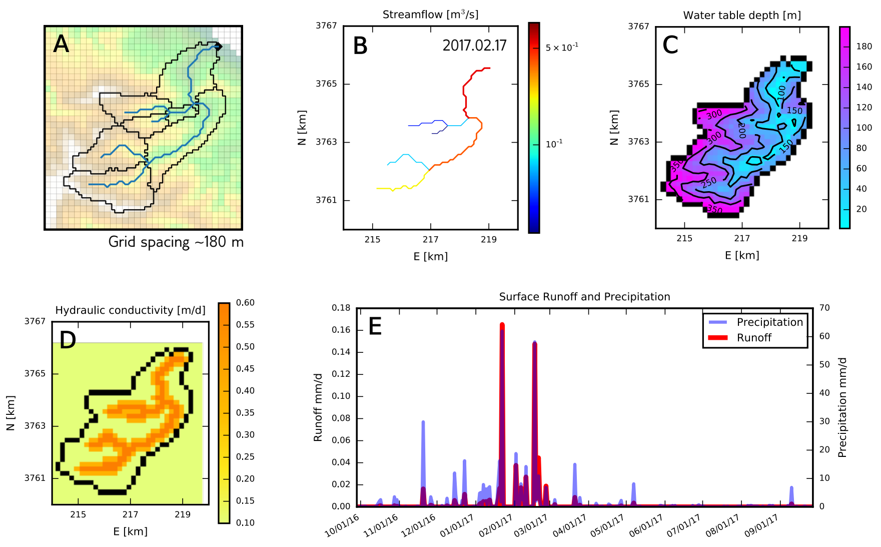

Figure 6. Water Canyon, Santa Rosa Island, California, USA. **(A)** Map with MODFLOW grid, HRU outlines, stream segments (blue), and digital elevation model. **(B)** Streamflow through the drainage network is typically less than 1 m^3/s in this semi-arid system. **(C)** The water table is deepest below ridge tops and becomes shallow in the valleys. **(D)** We simulate hydraulic conductivity as increasing around the channel network to represent alluvium and colluvium. **(E)** Surface runoff contributions to catchment-wide discharge correlate in time with precipitation, but are much lower in magnitude. Under the semi-arid climate forcing, the model simulates most rainfall infiltrating to recharge the aquifer, with relatively little overland flow. This result likely underestimates actual surface runoff, considering the the significant erosive overland flow events have occurred in the recent past (Wagner et al., 2004).

Here we apply the GRASS-GSFLOW toolkit to model Water Canyon, one of the island's many drainages. The topographic relief in this example is intermediate between the Shullcas and Cannon River watersheds, and it is unique among our three sites in that its outflow drains to the ocean and that its semiarid climate leads to losing streams that may run dry (Jazwa et al., 2016). We drove this hydrologic model with topography derived from a 3 arcsecond SRTM DEM (Farr et al., 2007) and projected to a UTM coordinate system at 90 m resolution. This DEM, which was the basis for surface-water routing calculations in the GRASS GIS module, is the coarsest of those in any of our examples. Based on user-provided inputs, these cells were downsampled by a factor of two to 180-meter resolution to develop the MODFLOW grid; the degree of downsampling is chosen for computational efficiency, which is of little concern in this small and coarse-resolution catchment. Basic weather data (rainfall, temperature, and humidity) were obtained from the Western Regional Climate Center (wrcc.dri.edu), converted from their raw hourly frequency to the required daily inputs. This simple example in Figure 6 demonstrates GSFLOW-GRASS's capabilities in representing heterogeneous aquifer properties and probing surface hydrologic controls on erosion.

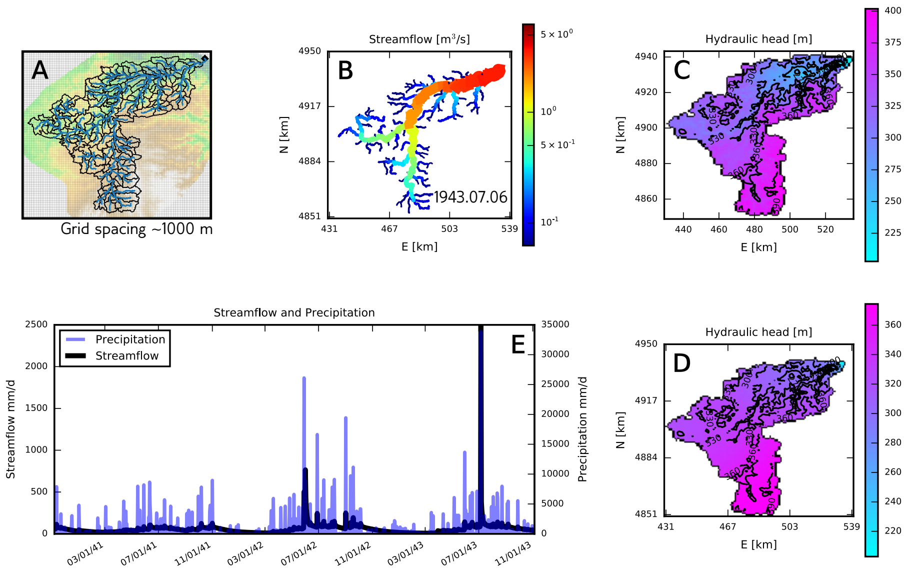

Figure 7. Cannon River, Minnesota, USA. **(A)** Map with MODFLOW grid, HRU outlines, stream segments (blue), and digital elevation model. **(B)** Discharge on the Cannon River after an 11 cm rainfall event. **(C, D)** Two MODFLOW layers were implemented to represent an upper glacial till unit and the underlying fractured carbonate bedrock with likely higher hydraulic conductivity. This generally low-relief catchment exhibits a shallow water table, except around the river gorge near the outlet. **(E)** The hydrograph shows the major 1943 storm event and its associated discharge, as well as the two years prior.

4.3 Cannon River, Minnesota, USA

The Cannon River is a tributary to the upper Mississippi River in Minnesota, USA. Its headwaters cross low-relief uplands that are capped by low-hydraulic-conductivity glacial deposits (Patterson and Hobbs, 1995), and are intensively farmed (Kreiling and Houser, 2016). Its lower reaches pass through a valley cut into fractured carbonate bedrock that is popular for recreation. This combination of agricultural and recreational use, and its transient geomorphology (low-gradient headwaters above a high-gradient river) are common in the formerly-glaciated Upper Midwest (Blumentritt et al., 2009; Carson et al., 2017). This leads to a suite of management concerns related to agricultural nutrients and fine sediments, and their interactions with both the surface water and the bedrock groundwater systems that underlie them (Tipping, 2006; Steenberg et al., 2013), thus motivating the need for integrated hydrologic modeling tools.

The Cannon River watershed covers 3720 km^2, with only 215 m of total relief, making it the largest and lowest-gradient of these study areas. Its average elevation is 340 m above sea level. For its surface topography, we use the Minnesota state-wide 1 m LiDAR data set (http://www.mngeo.state.mn.us/chouse/elevation/lidar.html), which we resampled to 15 m resolution. We discretize the subsurface of the Cannon River watershed into 1 km MODFLOW grid cells, and define new stream segments and HRUs for each tributary that has a drainage area exceeding 9 km^2. Meteorological data from nearby Zumbrota, Minnesota was obtained from the Midwestern Regional Climate Center (Adresen et al., 2014). The northern, mid-continental setting makes the Cannon River watershed the example with the most seasonally distributed precipitation and strongest seasonal temperature differences. GSFLOW outputs such as those shown in Figure 7 can be used to evaluate the susceptibility of shallow and low-gradient water tables to overlying agricultural inputs, which can also be episodically flushed into river channels during major storms.

5 Conclusions

We have created GSFLOW-GRASS, an easy-to-use toolkit for implementing the integrated surface-water and groundwater model GSFLOW using open-source Python scripts and GRASS GIS commands. This allow users equipped with a DEM, precipitation and temperature data, and basic knowledge about land-surface and subsurface properties to efficiently construct watershed-scale hydrologic simulations. This tool improves the accessibility of complex hydrologic software and will thus broaden the reach of integrated hydrologic models and their usage in both scientific research and practical resource management. We designed GSFLOW-GRASS to strike a balance between functioning quickly out-of-the-box and including full flexibility for customizing model runs. A default implementation can be launched with no programming required by the user, while all toolkit scripts are commented with instructions for adjusting any model inputs if desired. Other key features of the toolkit include the use of all open-source software, enabling users anywhere to apply GSFLOW, and the implementation of topologically based domain discretizations, which may afford computational advantages for certain applications. GSFLOW-GRASS complements a software package recently released by the USGS that uses ArcGIS tools to set up regular grid domains for GSFLOW (Gardner et al., 2017). Providing different model implementation options affords greater flexibility depending on a user's software preferences and data formats. The coincident release of these two modeling support packages opens opportunities for rigorous testing of different model discretization and implementation strategies that can benefit different research and resource management applications. We have demonstrated GSFLOW-GRASS with three diverse examples based on topographies and climates from the water-stressed Andes, Santa Rosa Island off the coast of California, USA, and the intensely farmed upper midwest region of the United States. The results show that GSFLOW-GRASS offers a flexible tool for investigating the role groundwater-surface water interactions in modulating dry season discharge, controlling runoff in erosion-prone landscapes, and imposing possible water quality threats in agricultural and recreational watersheds.

Code availability. The version of GSFLOW-GRASS used for this paper is available at https://github.com/UMN-Hydro/GSFLOW-GRASS/releases. Updated versions of our code are downloadable directly from the UMN-Hydro repository on GitHub, at https://github.com/UMN-Hydro/GSFLOW-GRASS. The user's manual is available as the README.md file in the repository. The GSFLOW executable and source code are available in the UMN-Hydro repository https://github.com/UMN-Hydro/GSFLOW-1.2.0 and from the USGS website https://water.usgs.gov/ogw/gsflow/. GRASS GIS 7.3+ is available from https://grass.osgeo.org/download/software/.

Author contributions. G.-H. C. Ng and A. D. Wickert together conceived of the project idea, wrote the majority of the paper, and built the visualization scripts. G.-H. C. Ng wrote the input file builder and plotting scripts. A. D. Wickert built the GRASS GIS extensions and integrated them with Ng's workflow. L. D. Somers and C. Cronkite-Ratcliff tested the code during its development and contributed their study cases. L. Saberi helped code the input-file builder scripts. R. Niswonger provided advice and support from the U.S. Geological Survey team that develops GSFLOW. J. M. McKenzie provided support for the Shullcas test case.

Competing interests. The authors declare that they have no conflict of interest.

Acknowledgements. The authors acknowledge all contributors to MODFLOW, PRMS, and GSFLOW, without whom this work would not be possible.

References

Adresen, J., Hilberg, S., and Kunkel, K.: Historical climate and climate trends in the Midwestern United States, Climate Change in the Midwest: A Synthesis Report for the National Climate Assessment, pp. 8–36, 2014.

Ajami, N. K., Gupta, H., Wagener, T., and Sorooshian, S.: Calibration of a semi-distributed hydrologic model for streamflow estimation along a river system, Journal of Hydrology, 298, 112–135, https://doi.org/10.1016/j.jhydrol.2004.03.033, 2004.

Arnold, J. G. and Fohrer, N.: SWAT2000: Current capabilities and research opportunities in applied watershed modelling, Hydrological Processes, 19, 563–572, https://doi.org/10.1002/hyp.5611, 2005.

Arroyo Aliaga, J., Gurmendi Párraga, P., and Machuca Manrique, E.: Efectos de las anom alías climáticas en la cobertura de nieve de los glaciares centrales del Perú, Apuntes de Ciencia & Sociedad, 5, 146–156, file:///C:/Users/ISABEL/Downloads/310-1312-4-PB.pdf, 2015.

Bandaragoda, C., Tarboton, D. G., and Woods, R.: Application of TOPNET in the distributed model intercomparison project, Journal of Hydrology, 298, 178–201, https://doi.org/10.1016/j.jhydrol.2004.03.038, http://linkinghub.elsevier.com/retrieve/pii/S0022169404002446, 2004.

Blumentritt, D. J., Wright, H. E., and Stefanova, V.: Formation and early history of Lakes Pepin and St. Croix of the upper Mississippi River, Journal of Paleolimnology, 41, 545–562, 2009.

Butts, M. and Graham, D.: Flexible Integrated Watershed Modeling with MIKE SHE, in: Watershed Models, pp. 245–271, CRC Press, https://doi.org/10.1201/9781420037432.ch10, http://www.crcnetbase.com/doi/10.1201/9781420037432.ch10, 2005.

Carson, E. C., Rawling III, J. E., Attig, J. W., , and Bates, B. R.: Late Cenozoic evolution of the upper Mississippi River, stream piracy, and reorganization of North American drainage systems, GSA Today, accepted, 2017.

Clark, R. A., Halvorson, W. L., Sawdo, A. A., and Danielsen, K. C.: Plant communities of Santa Rosa Island, Channel Islands National Park, vol. Technical, University of California, Davis, California, 1990.

Czuba, J. A. and Foufoula-Georgiou, E.: A network-based framework for identifying potential synchronizations and amplifications of sediment delivery in river basins, Water Resources Research, 50, 3826–3851, https://doi.org/10.1002/2013WR014227, 2014.

Czuba, J. A. and Foufoula-Georgiou, E.: Dynamic connectivity in a fluvial network for identifying hotspots of geomorphic change, Water Resources Research, 51, 1401–1421, https://doi.org/10.1002/2014WR016139, 2015.

Doherty, J.: PEST: A Unique Computer Program for Model-independent Parameter Optimisation, in: Water Down Under 94: Groundwater/Surface Hydrology Common Interest Papers; Preprints of Papers, p. 551, http://search.informit.com.au/documentSummary;dn=752715546665009;res=IELENG, 1994.

Essaid, H. I. and Hill, B. R.: Watershed-scale modeling of streamflow change in incised montane meadows, Water Resources Research, 50, 2657–2678, https://doi.org/10.1002/2013WR014420, 2014.

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., and Alsdorf, D.: The Shuttle Radar Topography Mission, Reviews of Geophysics, 45, RG2004, https://doi.org/10.1029/2005RG000183, http://doi.wiley.com/10.1029/2005RG000183, 2007.

Feldman, A. D.: Hydrologic modeling system HEC-HMS, Technical Reference Manual, Technical Reference Manual, p. 145, https://doi.org/CDP-74B, http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Hydrologic+Modeling+System+HEC-HMS#7, 2000.

Galeone, D. G., Risser, D. W., Eicholtz, L. W., and Hoffman, S. A.: Water Quality and Quantity and Simulated Surface-Water and Groundwater Flow in the Laurel Hill Creek Basin, Southwestern Pennsylvania, 1991-2007, vol. 2016-5082 of *Scientific Investigations Report*, U.S. Geological Survey, https://doi.org/10.3133/sir20165082, 2016.

Gannett, M., Lite, K. J., Risley, J., Pischel, E., and La Marche, J.: Simulation of Groundwater and Surface-Water Flow in the Upper Deschutes Basin, Oregon, vol. 2017–5097 of *Scientific Investigations Report*, U.S. Geological Survey, https://doi.org/10.3133/sir20175097, 2017.

Gardner, M., Morton, C., Huntington, J. L., Niswonger, R. G., and Henson, W. R.: Input Data Processing Tools for Integrated Hydrologic Models, Environmental Modelling and Software, Submitted, 2017.

Gómez, G. C., Cerrón, R. M., Capcha, T. M., and Villavicencio, C. O.: Evaluación de la Tasa de Infiltración en Tierras Agrícolas, Forestales y de Pastoreo en la Subcuenca del Río Shullcas, Apuntes de Ciencia & Sociedad, 4, 32–43, 2014.

Hansen, A. T., Czuba, J. A., Schwenk, J., Longjas, A., Danesh-Yazdi, M., Hornbach, D. J., and Foufoula-Georgiou, E.: Coupling freshwater mussel ecology and river dynamics using a simplified dynamic interaction model, Freshwater Science, 35, 200–215, https://doi.org/10.1086/684223, http://www.jstor.org/stable/10.1086/684223, 2016.

Harbaugh, A. W.: MODFLOW-2005 , The U . S . Geological Survey Modular Ground-Water Model — the Ground-Water Flow Process, in: Book 6. Modeling techniques, Section A. Ground Water, vol. U.S. Geolo, chap. 16, p. 253, U.S. Geological Survey, Reston, Virginia, USA, https://doi.org/U.S. Geological Survey Techniques and Methods 6-A16, 2005.

Hassan, S. T., Lubczynski, M. W., Niswonger, R. G., and Su, Z.: Surface–groundwater interactions in hard rocks in Sardon Catchment of western Spain: An integrated modeling approach, Journal of Hydrology, 517, 390–410, https://doi.org/10.1016/j.jhydrol.2014.05.026, http://dx.doi.org/10.1016/j.jhydrol.2014.05.026http://linkinghub.elsevier.com/retrieve/pii/S0022169414003904, 2014.

Hofierka, J., Mitášová, H., and Neteler, M.: Chapter 17 Geomorphometry in GRASS GIS, in: Developments in Soil Science, vol. 33, chap. 17, pp. 387–410, https://doi.org/10.1016/S0166-2481(08)00017-2, http://linkinghub.elsevier.com/retrieve/pii/S0166248108000172, 2009.

Hsieh, P. A. and Winston, R. B.: User's Guide To Model Viewer, a Program for Three-Dimensional Visualization of Ground-Water Model Results, vol. 02-106 of *Open-File Report*, U.S. Geological Survey, Menlo Park, CA, 2002.

Hunt, R. J., Walker, J. F., Selbig, W. R., Westenbroek, S. M., and Regan, R. S.: Simulation of Climate - Change effects on streamflow, Lake water budgets, and stream temperature using GSFLOW and SNTEMP, Trout Lake Watershed, Wisconsin, USGS Scientific Investigations Report., pp. 2013–5159, 2013.

Hunter, J. D.: Matplotlib: A 2D graphics environment, Computing in Science and Engineering, 9, 99–104, https://doi.org/10.1109/MCSE.2007.55, 2007.

Jasiewicz, J. and Metz, M.: A new GRASS GIS toolkit for Hortonian analysis of drainage networks, Computers and Geosciences, 37, 1162–1173, https://doi.org/10.1016/j.cageo.2011.03.003, 2011.

Jazwa, C. S., Duffy, C. J., Leonard, L., and Kennett, D. J.: Hydrological Modeling and Prehistoric Settlement on Santa Rosa Island, California, USA, Geoarchaeology, 31, 101–120, https://doi.org/10.1002/gea.21532, 2016.

Kreiling, R. M. and Houser, J. N.: Long-term decreases in phosphorus and suspended solids, but not nitrogen, in six upper Mississippi River tributaries, 1991–2014, Environmental Monitoring and Assessment, 188, https://doi.org/10.1007/s10661-016-5464-3, http://dx.doi.org/10.1007/s10661-016-5464-3, 2016.

Leavesley, G. H., Lichty, R., Troutman, B., and Saindon, L.: Precipitation-Runoff Modeling System: User's Manual, vol. Water Reso, U.S. Geological Survey, Denver, Colorado, USA, 1983.

Leavesley, G. H., Restrepo, P., Markstrom, S., Dixon, M., and Stannard, L.: The Modular Modeling System (MMS): User´s Manual, vol. 1, 1996.

Leopold, L. B. and Maddock, T.: The hydraulic geometry of stream channels and some physiographic implications, Professional Paper, United States Geological Survey, Washington, D.C., 1953.

López-Moreno, J., Fontaneda, S., Bazo, J., Revuelto, J., Azorin-Molina, C., Valero-Garcés, B., Morán-Tejeda, E., Vicente-Serrano, S., Zubieta, R., and Alejo-Cochachín, J.: Recent glacier retreat and climate trends in Cordillera Huaytapallana, Peru, Global and Planetary Change, 112, 1–11, https://doi.org/10.1016/j.gloplacha.2013.10.010, http://linkinghub.elsevier.com/retrieve/pii/S0921818113002385, 2014.

Markstrom, S. L., Niswonger, R. G., Regan, R. S., Prudic, D. E., and Barlow, P. M.: GSFLOW—Coupled Ground-Water and Surface-Water Flow Model Based on the Integration of the Precipitation-Runoff Modeling System (PRMS) and the Modular Ground-Water Flow Model (MODFLOW-2005), U.S. Geological Survey, p. 240, https://doi.org/10.13140/2.1.2741.9202, http://pubs.er.usgs.gov/publication/tm6D1, 2008.

Markstrom, S. L., Regan, R. S., Hay, L. E., Viger, R. J., Webb, R. M. T., Payn, R. A., and LaFontaine, J. H.: PRMS-IV , the Precipitation-Runoff Modeling System, Version 4, U.S. Geological Survey, Reston, Virginia, USA, https://doi.org/http://dx.doi.org/10.3133/tm6B7, 2015.

Maxwell, R., Putti, M., Meyerhoff, S., Delfs, J., Ferguson, I. M., Ivanov, V., Kim, J., Kolditz, O., Kollet, S. J., Kumar, M., Lopez, S., Niu, J., Paniconi, C., Park, Y., Phanikumar, M., Shen, C., Sudicky, E. A., and Sulis, M.: Water Resources Research, Water resources research, 50, 1531–1549, https://doi.org/10.1002/2013WR013725.Received, 2014.

Metz, M., Mitasova, H., and Harmon, R. S.: Efficient extraction of drainage networks from massive, radar-based elevation models with least cost path search, Hydrology and Earth System Sciences, 15, 667–678, https://doi.org/10.5194/hess-15-667-2011, http://www.hydrol-earth-syst-sci.net/15/667/2011/, 2011.

Minnich, R. A.: Grazing, fire, and the management of vegetation on Santa Catalina Island, California, Tech. rep., Pacific Southwest Forest and Range Experiment Station, Forest Service, U.S. Department of Agriculture, Berkeley, California, USA, http://www.fs.fed.us/psw/publications/documents/psw_gtr058/psw_gtr058_6a_minnich.pdf, 1982.

Mitasova, H., Mitas, L., Brown, W. M., Gerdes, D. P., Kosinovsky, R., and Baker, T.: Modelling spatially and temporally distributed phenomena: New methods and tools for GRASS GIS, International Journal of Geographical Information Systems, 9, 433–446, https://doi.org/10.1080/02693799508902048, 1995.

Neitsch, S., Arnold, J., Kiniry, J., Srinivasan, R., and Williams, J.: Soil and Water Assessment Tool User's Manual, vol. TR-192, Texas Water Resources Institute, College Station, Texas, USA, http://swat.tamu.edu/media/1294/swatuserman.pdf, 2002.

Neteler, M. and Mitasova, H.: Open Source GIS: A GRASS GIS Approach, Springer, New York, New York, USA, third edit edn., 2008.

Neteler, M., Beaudette, D., Cavallini, P., Lami, L., and Cepicky, J.: GRASS GIS, Open Source Approaches in Spatial Data Handling, 2, 171–199, https://doi.org/10.1007/978-3-540-74831-1_9, http://dx.doi.org/10.1007/978-3-540-74831-1_9, 2008.

Neteler, M., Bowman, M. H., Landa, M., and Metz, M.: GRASS GIS: A multi-purpose open source GIS, Environmental Modelling & Software, 31, 124–130, https://doi.org/10.1016/j.envsoft.2011.11.014, http://dx.doi.org/10.1016/j.envsoft.2011.11.014http://linkinghub.elsevier.com/retrieve/pii/S1364815211002775, 2012.

Niswonger, R. G., Panday, S., and Motomu, I.: MODFLOW-NWT, A Newton Formulation for MODFLOW-2005, vol. Techniques, U.S. Geological Survey, Reston, Virginia, USA, 2011.

Pal, J. S., Giorgi, F., Bi, X., Elguindi, N., Solmon, F., Gao, X., Rauscher, S. A., Francisco, R., Zakey, A., Winter, J., Ashfaq, M., Syed, F. S., Bell, J. L., Differbaugh, N. S., Karmacharya, J., Konari, A., Martinez, D., Da Rocha, R. P., Sloan, L. C., and Steiner, A. L.: Regional climate modeling for the developing world: The ICTP RegCM3 and RegCNET, Bulletin of the American Meteorological Society, 88, 1395–1409, https://doi.org/10.1175/BAMS-88-9-1395, 2007.

Patterson, C. J. and Hobbs, H. C.: Surficial geology, in: County Atlas C-9: Geologic atlas of Rice County, edited by Hobbs, H. C., vol. pl. 3 of *1:100,000*, Minnesota Geological Survey, Saint Paul, Minnesota, USA, 1995.

Pérez, F. and Granger, B. E.: IPython: A system for interactive scientific computing, Computing in Science and Engineering, 9, 21–29, https://doi.org/10.1109/MCSE.2007.53, 2007.

Poeter, E. P. and Hill, M. C.: Documentation of UCODE, A Computer Code for Universal Inverse Modeling, USGS Water Resources Investigations Report 98-4080, vol. Water-Reso, U.S. Geological Survey, Denver, Colorado, USA, 1998.

Poeter, E. P. and Hill, M. C.: UCODE, a computer code for universal inverse modeling, Computers and Geosciences, 25, 457–462, https://doi.org/10.1016/S0098-3004(98)00149-6, 1999.

Qu, Y. and Duffy, C. J.: A semidiscrete finite volume formulation for multiprocess watershed simulation, Water Resources Research, 43, 1–18, https://doi.org/10.1029/2006WR005752, 2007.

Reed, S., Koren, V., Smith, M., Zhang, Z., Moreda, F., and Seo, D. J.: Overall distributed model intercomparison project results, Journal of Hydrology, 298, 27–60, https://doi.org/10.1016/j.jhydrol.2004.03.031, 2004.

Regan, R. S., Niswonger, R. G., Markstrom, S. L., and Barlow, P. M.: Documentation of a restart option for the U.S. Geological Survey coupled groundwater and surface-water flow (GSFLOW) model, vol. 6-D3, U.S. Geological Survey, Reston, Virginia, USA, 2015.

Reilly, T. E.: System and Boundary Conceptualization in Ground-Water Flow Simulation, in: Techniques of Water-Resources Investigations of the United States Geological Survey, Book 3, Applications of Hydraulics, chap. B8, p. 38, U.S. Geological Survey, 2001.

Reilly, T. E. and Harbaugh, A. W.: Guidelines for Evaluating Ground-Water Flow Models, vol. 2004-5038, U.S. Geological Survey, 2004.

Shapiro, M. and Westervelt, J.: r. mapcalc: An algebra for GIS and image processing, vol. USACERL-TR, Construction Engineering Research Lab, Champaign, Illinois, USA, 1994.

Simunek, J., Sejna, M., Saito, H., Sakai, M., and van Genuchten, M.: "The HYDRUS-1D software package for simulating the one-dimensional movement of water, heat, and multiple solutes in variably-saturated media. Version 4.08. HYDRUS Softw. Ser. 3., Tech. Rep. January, 2009.

Steenberg, J. R., Tipping, R. G., and Runkel, A. C.: Geologic controls on groundwater and surface water flow in southeastern Minnesota and its impact on nitrate concentrations in streams, Minnesota Geological Survey Open File Report, p. 154, 2013.

Surfleet, C. G. and Tullos, D.: Uncertainty in hydrologic modelling for estimating hydrologic response due to climate change (Santiam River, Oregon), Hydrological Processes, 27, 3560–3576, https://doi.org/10.1002/hyp.9485, http://doi.wiley.com/10.1002/hyp.9485, 2013.

Tachikawa, T., Kaku, M., Iwasaki, A., Gesch, D., Oimoen, M., Zhang, Z., Danielson, J., Krieger, T., Curtis, B., Haase, J., Abrams, M., Crippen, R., Carabajal, C., and ASTER GDEM Validation Team: ASTER Global Digital Elevation Model Version 2 – Summary of Validation Results, Tech. rep., NASA Earth Resources Observation and Science (EROS) Center, Sioux Falls, South Dakota, USA, https://doi.org/10.1017/CBO9781107415324.004, http://www.jspacesystems.or.jp/ersdac/GDEM/ver2Validation/Summary_GDEM2_validation_report_final.pdf, 2011.

Tian, Y., Zheng, Y., Wu, B., Wu, X., Liu, J., and Zheng, C.: Modeling surface water-groundwater interaction in arid and semi-arid regions with intensive agriculture, Environmental Modelling and Software, 63, 170–184, https://doi.org/10.1016/j.envsoft.2014.10.011, 2015.

Tipping, R. G.: Subsurface recharge and surface infiltration, in: Geologic Atlas of Scott County, Minnesota, Minnesota Geological Survey Atlas Series, 2006.

Travezan Adauto, D.: Disponibilidad de pago por los agricultores para la conservación de los recursos hídricos en la microcuenca del Río Shullcas-Huancayo 2014, Ph.D. thesis, Universidad Nacional del Centro del Perú, Huancayo, 2015.

Viger, R. J. and Leavesley, G. H.: The GIS Weasel User's Manual, U.S. Geological Survey, 2007.

Vivoni, E. R., Ivanov, V. Y., Bras, R. L., and Entekhabi, D.: Generation of Triangulated Irregular Networks Based on Hydrological Similarity, Journal of Hydrologic Engineering, 9, 288–302, https://doi.org/10.1061/(ASCE)1084-0699(2004)9:4(288), 2004.

Wagner, J., Martin, M., Faulkner, K. R., Chaney, S., Noon, K., Denn, M., and Reiner, J.: Riparian System Recovery After Removal of Livestock From Santa Rosa Island, Channel Islands National Park, California, Tech. rep., National Park Service Technical Report NPS/NRWRD/NRTR-2004/324, Fort Collins, Colorado, USA, 2004.

Waterloo Hydrogeologic Inc.: Visual MODFLOW 2011.1 User's Manual: For Professional Applications in Three-Dimensional Groundwater Flow and Contaminant Transport Modeling, Waterloo Hydrologic, Inc., Waterloo, Ontario, Canada, http://trials.swstechnology.com/software/Visual_MODFLOW/2011/Manuals_and_Guides/VMOD-2011.1_Manual.pdf, 2011.

Winston, R. B.: Graphical User Interface for MODFLOW, Version 4, vol. 00-315 of *Open-File Report*, U.S. Geological Survey, Reston, Virginia, USA, 2000.

Winston, R. B.: ModelMuse: A Graphical User Interface for MODFLOW-2005 and PHAST, in: Section A, Ground Water—Book 6, Modeling Techniques, vol. 6-A29 of *Techniques and Methods*, chap. 29, p. 52 p, U.S. Geological Survey, Reston, Virginia, USA, http://pubs.usgs.gov/tm/tm6A29/tm6A29.pdf, 2009.

Winter, T. C., Harvey, J. W., Franke, O. L., and Alley, W. M.: Ground water and surface water: A single resource, U.S. Government Printing Office, Denver, Colorado, USA, 1998.

Zambelli, P., Gebbert, S., and Ciolli, M.: Pygrass: An Object Oriented Python Application Programming Interface (API) for Geographic Resources Analysis Support System (GRASS) Geographic Information System (GIS), ISPRS International Journal of Geo-Information, 2, 201–219, https://doi.org/10.3390/ijgi2010201, http://www.mdpi.com/2220-9964/2/1/201/, 2013.

Ŝúri, M. and Hofierka, J.: A new GIS-based solar radiation model and its application to photovoltaic assessments, Transactions in GIS, 8, 175–190, https://doi.org/10.1111/j.1467-9671.2004.00174.x, 2004.