All of the referee's review comments are copied below and shaded in gray. Our responses are in italics. New text is quoted and/or referenced in boldface.

I would like to thank the authors for their effort in clarifying several points made in my last round of review comments. In acknowledging such effort, I must recognize that my major concerns, expressed during the first and second iteration of this review process, still persist. These points cannot be addressed just reformulating some sentences in the text but they need, in my opinion, additional substantial work by the authors. Indeed, as previously stated, the Results section fails to offer a clear view of the technical advancements proposed with the GSFLOW-GRASS toolkit. This is, in my opinion, a major issue for a GMD paper. Here I must also highlight that despite this central concern, Figures 5-6-7 (and their information content) have remained nearly unchanged through the review process. I will try to re-iterate on these major issues starting from the newly included table (i.e., Table 2 of the revised manuscript) that summarizes the tests of GSFLOW-GRASS capabilities/technical challenges for the three selected test cases.

We greatly appreciate the referee's continued attention and time to ensure that our manuscript is of suitable quality for GMD, and we are glad that the referee recognized improvements in our last revision. We are of course disappointed that the referee still finds major concerns, as we did seriously consider all previous comments. In this revision, we focus on three points that we anticipate should satisfactorily address the latest comments.

(1)
First and foremost, we address the referee's main concern about our toolbox's technical merits by substantiating with specific details the statements flagged in our 3 examples. Despite our previous efforts to clarify technical advancements, the referee's persistent comments made us realize that indeed, a number of our claims remained vague and would greatly benefit from direct support. We added precise comparison numbers, detailed literature examples, and improved explanations in the three example in Section 4, as detailed in our specific responses below.

Further, we substantially edited Section 3.2 ("GRASS GIS domain builder") to more clearly highlight the overall technical contributions. We had previously left some technical details vague in the examples section because we had explained the general points in Section 3.2 (GRASS-GIS Domain Builder). However, we now see that we had not adequately highlighted the innovations in Section 3.2, and further, the section was not well-organized or referenced in the examples for the reader to easily tie them together. We now provide new sub-section headings for easier references to Section 3.2. ***We have almost completely rewritten the part on the surface-water network (now Section 3.2.1)*** *to clearly detail the computational advantages of GRASS-GIS's r.watershed extension, supported by a complete literature review; then, to address the reviewer's concern that using an existing software is an insufficient technical advancement (see below), we again provide a complete literature review to explain that ours is the first effort to incorporate this algorithm into an integrated hydrologic model toolbox, thus newly harnessing its power in this field. Other changes to Section 3.2.1 serve to more clearly explain the different features of the surface-water network builder.* ***The entire section on the Groundwater-flow grid (now Section 3.2.2) was also significantly edited****, in large part to better explain the hydrologic correction of coarsened MODFLOW grids - an aspect in the examples that the referee was unable to follow in the previous version (see below).* ***Finally, we also made major edits to the section on support for additional GSFLOW***

functionality (now Section 3.2.3). *We added the new extension "v.gsflow.mapdata" in our previous revision in response to reviewer comments about the limited input data support provided by GSFLOW-GRASS compared to some other integrated hydrologic modeling pre-processing packages. This revision includes more details on the types of customized inputs that users can make with this new extension.*

Please see Section 3.2 (p. 10) - in particular, 3.2.1 (p. 11), 3.2.2 (p. 13), and 3.2.3 (p. 13). *Those sections are almost entirely re-written, but we do not attempt to copy them here because of their length.*

Finally, related to the reviewer's concern about Figures 5-7, we would like to point out that we did in fact modify Figure 7 in our previous revision to include a comparison with observed streamflow in Cannon River. This was done to build confidence that our toolbox produces realistic preliminary simulations that can be further calibrated for specific study.

(2)
We acknowledge that our toolbox builds upon mostly existing algorithms and often straightforward programming rather than formulating brand-new methodologies, but we would like to argue that the primary technical innovation is to bring together these different components in an automated, accessible, and self-consistent package for integrated hydrologic modeling - a task that is not at all trivial and has not been previously done (specifically, using the powerful least-cost path flow-routing algorithm with integrated hydrologic modeling). As we point out in the manuscript, providing a practical utility that enables more researchers and practitioners to implement integrated hydrologic models can help accelerate advances in hydrologic science understanding and water resource management - we anticipate this to be our major contribution. We do appreciate that the referee wishes to uphold a high technical standard for GMD and we emphasize that we do NOT wish to water-down the technical quality of GMD articles, but we also believe that technical advancement can come in the form of innovatively integrating disparate components into a new practical and useful tool.

To check whether our perspective is consistent with the stated goals and requirements of GMD, we carefully reviewed the GMD editorial document (Hargreaves et al. 2015 - full reference is below) for guidance. We appreciate that the referee acknowledged in an earlier review that our manuscript does fit within GMD's call; we would like to further share the following excerpts.
- Support for contributions that are accessible and practical utilities:

> *"The papers should be detailed, complete, rigorous, and accessible to a wide community of geoscientists. In addition to complete models, this type of paper may also describe model components and modules, as well as frameworks and utility tools used to build practical modelling systems, such as coupling frameworks or other software toolboxes with a geoscientific application."*

- Support that the scientific goal need not be a major scientific/technical discovery:

> *"The scientific goal is reproducibility: ideally, the description should be sufficiently detailed to in principle allow for the re-implementation of the model by others, so all technical details which could substantially affect the numerical output should be described."*
>
> *"It is not expected from a GMD paper that it contain novel scientific discoveries."*

We acknowledge that "scientific goal / discoveries" do not strictly equate with "technical discovery," but in this editorial document, those were the closest terms we could find related to *"technical advancement"; in fact, the word "technical" appears only one time (cited above) in the entire section on "Model description papers" (Section A2) - the category of our manuscript. No where in the editorial is there a statement about a required level of technical innovation. As such, we do believe that after clarifying the technical aspects of our toolbox (the referee rightfully pointed out that our explanations needed to be improved), the degree of technical discovery should not be a metric used against our submission to GMD. Our toolbox already has proven accessibility and practical application based on GitHub statistics; outside our immediate collaborators, we have had:*

- *344 visitors since April 22, 2018, when we started to track this (~3/day)*
- *2 forks (dynamic copies for further code development by third parties)*
- *4 stars (bookmarks)*
- *1 external scientist actively using and building upon GSFLOW-GRASS after finding it online (his fork is one of the two)*

<u>Reference</u>

J. C. Hargreaves, A. Kerkweg, R. Marsh, A. Ridgwell, D. M. Roche, and R. Sander, "Editorial: The publication of geoscientific model developments v1.1," Geosci. Model Dev., vol. 8, no. 10, pp. 3487–3495, Oct. 2015.

(3)

We would like to clarify that only one purpose of the examples is to demonstrate technical aspects; they also serve to demonstrate applications with GSFLOW that can be facilitated with our toolbox. We feel that in our effort to clarify the technical points in response to the referee's previous comments, this other important purpose is becoming buried. Again, we share the following excerpts from Hargreaves et al. (2015) to support the appropriateness of using examples for this reason, and not solely for demonstrating technical advancements:

> *"The model description should be contextualised appropriately. For example, the inclusion of discussion of the scope of applicability and limitations of the approach adopted is expected."*
>
> *"Examples of model output should be provided, with evaluation against standard benchmarks, observations, and/or other model output included as appropriate."*

(Note that as mentioned above, we added streamflow observations to Figure 7 in the previous revision to show that out-of-the-box, results and plots from GSFLOW-GRASS can show realistic simulations as well as provide insight into how certain model parameters need to be further calibrated.) Some of the referee's comments made us realize that we were not always clear that some aspects of the examples served to highlight possible applications, and not necessarily technical advancements, and we have edited the text accordingly. ***We have also added the column "Applications" in Table 3 to summarize these applications for each example****. Specific details are provided below.*

The technical challenges identified for the first test case (i.e., Shullcas River, Peru) are the (i) steep topography and (ii) seasonal rainfall.

(i) As concerns the first technical challenge authors argue that a steep topography requires the definition of the surface flow path using high-resolution topography with the option of resolving the subsurface at relatively coarse resolution. Therefore, the technical challenge consists in handling a surface discretization not automatically aligned with subsurface flow path. This technical challenge reflects the sketch of Figure 1d, if I am right. Therefore, I think that a good way to test the toolkit would be to generate different subsurface configurations (in terms of grid resolution) with the same number of HRUs obtained with the high-resolution DEM (30 m for this test case). Here you could show the great potential of the automated workflow in generating consistent modeling scenarios that potentially subtend different level of information. Some concrete numbers on the computational gain would substantiate some too-vague statements still present in the manuscript ("…steep topography and narrow canyons of Shullcas is the need for impractically high resolution and computational expense if using a standard gridded model domain…")

We really appreciate the referee's comment, because it made clear that we inadequately explained the challenges of Shullcas and GSFLOW-GRASS' solutions for them. The steep topography and narrow canyons lead to a three different but related challenges, only one of which is the one that the referee was able to readily pick out (misaligned grids) based on how the previous version was written. The full list of challenges include: (1) need for a high-resolution surface flow representation that is computationally efficient, (2) need to link misaligned grids, and (3) need to coarsen MODFLOW regular grid without causing hydrologically incorrect flow. We have largely rewritten the Shullcas section to clarify this.

We also now see that our statements about how GSFLOW-GRASS addresses challenges were indeed vague. The referee had an excellent suggestion to provide concrete metrics to support our claims. The referee mentioned testing different grid alignments, but actually this was the most straightforward of the challenges and was solved with code that handles painstaking indexing between the domains, so there is not much to test. Instead, we decided it would be more meaningful to substantiate the other two solutions. To demonstrate the computational efficiency from the irregular grid, we now report that only 79 HRU cells were needed to capture nearly all of the essential flow-routing information from the original high-resolution 2x10^5 cells. ***We added a column with flow-routing grid cell size to Table 2 to facilitate the comparison****. To explain the value of hydrologically corrected MODFLOW cells in steep canyons, we now explain that without it, simulations of the Shullcas watershed showed water converging next to instead of within the stream channel network.*

The third paragraph of the Shullcas section was almost entirely rewritten to incorporate the above changes:

> ***(p. 23 line 11-30)***
> ***"The steep topography makes the Shullcas River Watershed an apt testbed for examining the ability of GSFLOW-GRASS to represent surface-water-groundwater links in challenging terrain. The major obstacle with Shullcas' steep topography and narrow canyons is the need for high resolution to represent surface flow, which leads to an impractically high number of computational units and expense if using a standard gridded model domain. An irregular surface grid can provide a much more efficient discretization, but this then entails painstaking indexing to link it to the subsurface grid, which must be a regular rectangular grid for the MODFLOW component of GSFLOW. Also, as the most computationally expensive part of GSFLOW, practical MODFLOW implementation typically needs a coarser resolution grid than that used for resolving the stream network, but simple coarsening of DEMs with steep gradients (e.g., by using mean elevations from the higher-resolution DEM) can result in hydrologically incorrect groundwater flow directions. GSFLOW--GRASS addresses these problems by computing flow paths***

using high-resolution topography (~2x10^5 grid cells of 900-m^2 size) and converting these into a far smaller number of larger computational surface cells (79 HRUs that are >=1 km^2 in area) that convey the same fundamental surface-flow information (see Table 2); this efficiency is possible because the surface is discretized along topographically defined surface-water hydrologic units -- stream segments and sub-basins (Section 3.2.1). To create a coarsened subsurface rectangular grid domain for the MODFLOW component, GSFLOW-GRASS' hydrological correction to enforce integrated subsurface drainage (Section 3.2.2) proved essential for preventing unrealistic results. Early model tests for Shullcas showed that simple grid coarsening using the mean value of the elevations from the higher-resolution grid could, for example, average elevations between flat valley floors and steep canyon walls. This caused cells containing the stream to be higher in elevations than the surrounding surface on the groundwater flow grid, leading to lateral flow out of these "dams" that formed as a numerical artifact of averaging. As the final step of the domain-building solution, GSFLOW-GRASS extensions seamlessly link the hydrologically corrected coarse-scale MODFLOW domain with the irregular surface HRUs (Section 3.2.3)."

(ii) The aim of this work is not to present any scientific validation of the results obtained by implementing the GSFLOW-GRASS workflow, as also stated by authors. Therefore, I do not see any technical challenge implied in the term "Seasonal rainfall" included by authors in this table.

This referee is correct that "seasonal rainfall" was inappropriately identified as a technical challenge. We realized that we were conflating the two different purposes of the examples: (1) demonstrating technical advances and capabilities of the toolbox and (2) demonstrating hydrological processes and applications possible with GSFLOW-GRASS. The point of "seasonal rainfall" should fall in the 2nd category - it is an example of a use case of GSFLOW-GRASS. ***We have now modified Table 2 so that it is split into two separate tables.*** *Table 2 contains only the numerical data for the watersheds, while Table 3 contains the "GSFLOW-GRASS Capabilities" and "Applications" demonstrated with each watershed example.* ***"Seasonal rainfall" is no longer placed under "GSFLOW-GRASS Capabilities" for Shullcas and is instead listed under "Applications" in Table 3.***

The technical challenges identified for the second test case (i.e., Water Canyon, USA) are the (i) NULL cells defining irregular coastline, (ii) small basin, and (iii) small number of coarse-resolution cells.

(i) I must admit that I still do not see the challenge of handling NULL cells. Authors show the modification of the code but this remains for me a basic feature of any terrain analysis tool.

The referee's comment made us realize that we did not adequately explain the problem with coastal boundaries. The referee is correct that assigning NULL values alone is not an issue; instead, the complication is in how to set boundary conditions for flow at a coastal pour-point where the NULL value is encountered. We rewrote and greatly expanded this section about the coastline to explain the need to carefully set constant head boundary conditions, and ***we have reworded the issue in Table 3 on "GSFLOW-GRASS capabilities" to refer to the irregular coastline boundary condition rather than to NULL values****. The revised main text reads:*

> ***(p. 25 line 7-15)***
> ***"Water Canyon is unique among the three example sites in that its outflow drains to the ocean. It therefore requires GSFLOW-GRASS to accommodate irregular boundaries (coastlines) by properly assigning boundary conditions and routing flow through them. Users identify ocean pixels by assigning NULL values to them; this causes flow routing from r.watershed to stop at the shoreline. To allow flow out of pour-point at the mouth of the river, the immediately downgradient MODFLOW cell can be set as a constant-head boundary, but this cell must be chosen carefully. The finite-difference scheme in MODFLOW dictates that the constant head boundary condition must be supplied along one of the four cardinal directions of the pour-point. Therefore, if the river flows diagonally to the sea, its constant-head boundary must be moved to the closest non-diagonal cell. v.gsflow.grid finds the proper constant-head boundary cell to set for the coastal case, as well as for any inland drainage case in which the pour point also requires a downgradient constant-head boundary."***

(ii) Why a small basin is a challenging case for GSFLOW-GRASS?
(iii) What's the technical challenge associated to a small number of coarse-resolution cells? What's the meaning of small?

Compared to the other two examples, Santa Rosa has both the coarsest resolution for determining the flow-routing and the smallest drainage area, which means that there is the sparsest amount of information to resolve how water is moving in the smallest watershed. That makes it the most difficult to accurately predict flow of the three examples, which is why we previously pointed out this aspect of Santa Rosa. However, we now realize this is more related to the choice of datasets and watersheds rather than the toolbox method, and so highlighting this "challenge" is a distraction to the reader - just as the referee found. ***We have removed any mention of the small drainage size and coarse flow-routing resolution from the Santa Rosa example*** *and instead now focus on the important features that the example showcases relevant to the toolbox capabilities. These include the NULL values (details above), and the hydrologically corrected downscaling method (details below).*

In addition to the technical capabilities demonstrated with Santa Rosa, we also expanded our discussion of the hydrological processes demonstrated with this example. As mentioned in our overall response, we wish to better emphasize that there is dual purpose to our examples, and showing technical capabilities is only one, and example applications is the other. We heavily edited the final paragraph to further explain the potential of using GSFLOW-GRASS to investigate and manage erosion through simulations of surface runoff, and we demonstrate how the toolbox's example hydraulic conductivity script was used to create heterogeneous conditions found with eroded watersheds:

> ***(p. 26 line 6-12)***
> ***"The Santa Rosa example demonstrates an application in which GSFLOW--GRASS can be used to investigate and manage erosion associated with hydrological conditions. Erosion of upland areas moves sediment downslope to the areas flanking the stream channel, which contains coarser-grained alluvial sediments. We represented this heterogeneity using a spatially distributed hydraulic conductivity field (Figure 6D) generated with the example model input script***

included in GSFLOW-GRASS (Section 3.3.3). Figure 6E demonstrates how the post-processing tools can be used to evaluate surface runoff, a driver of erosion on the island (Schumann et al. 2016). Simulations show precipitation events triggering surface runoff (Figure 6E), which could denude the hillslopes and transport eroded sediment through the drainage network (Figure 6B)."

We have filled out "GSFLOW-GRASS Capabilities" and "Applications" in Table 3 accordingly for the Santa Rosa example.

According to point (i), (ii), and (iii) I have problems in identifying the technical capabilities of the proposed workflow from this second test case. Again, there are confusing statements in the discussion of the results that do not bring much insight:

"The GSFLOW-GRASS implementation does simulate low streamflow as expected (Figure 6B), but the domain-builder generated sufficiently thick and hydrologically corrected MODFLOW cells that they maintain water and avoid any computational problems."

What do you mean with "sufficiently thick" and "hydrologically corrected" MODFLOW cells?

The referee's comment helped us realize the above sentence was poorly worded, and we needed to provide more detail, because the hydrologically corrected MODFLOW cells are among the key capabilities demonstrated with this example. We essentially rewrote this paragraph to explain why the hydrologically corrected downscaling method for MODFLOW grid cells is important for preventing MODFLOW cells from becoming incorrectly dry and posing numerical problems; we also separated out the confusing statement about "sufficiently thick" MODFLOW cells and clarified that we are talking about MODFLOW layer depth:

(p. 25 line 16-p. 26 line 5)

"Losing streams such as those in the steep and semi-arid Water Canyon catchment often run dry (Jazwa et al. 2016). If this causes MODFLOW cells to lose all of their water, GSFLOW will fail to numerically converge. Thus, the Water Canyon example also serves to demonstrate GSFLOW-GRASS' ability to prevent this problem by (1) incorporating MODFLOW-NWT, which uses a Newton-Raphson solver for increased stability (Niswonger et al. 2011); (2) allowing the user to specify an adequately deep MODFLOW discretization in the Settings file (Section 3.1) to supply sufficient water through the dry season; and (3) hydrologically correcting the elevations of coarsened MODFLOW cells to enforce integrated drainage through the stream network. Focusing on the third approach that is specific to the GSFLOW-GRASS toolbox (Section 3.2.2), the narrow and steep Water Canyon requires the same hydrologic corrections that were applied in the Shullcas case, above, to maintain downslope-integrated drainage. Under losing stream conditions, artificially increased channel elevation would steepen the hydraulic head gradient away from the channel and cause it to over-simulate water flow to the surrounding landscape. Therefore, hydrologic correction of the coarse MODFLOW grid is necessary to simulate appropriate head gradients and maintain water in cells, which is further required for any attempt to match stream-gauge records."

We have filled out "GSFLOW-GRASS capabilities" in Table 3 accordingly.

The technical challenges identified for the third test case are (i) two-layer soil hydraulic conductivity, (ii) poorly integrated drainage, (iii) low relief, and (iv) large basin.

(i) The tasks associated to the first point (i.e., two-layer soil hydraulic conductivity) are not discussed in the main text.

Thank you for pointing this out. We now mention the two-layer soil hydraulic conductivity in the main text:

> ***(p. 27 line 13-14)***
> ***"The flexible GSFLOW-GRASS input builder allows for easy implementation of two MODFLOW layers to represent an upper glacial till unit (low hydraulic conductivity) and the underlying fractured carbonate bedrock (higher hydraulic conductivity)."***

*We also **slightly shortened the caption text about it**, now that it is in the main text.*

(ii) and (iii) In my previous review I highlighted the problem that all the discussion for this test case is about an already existing GRASS feature (r.watershed). Therefore, one could argue that nothing new related to GSFLOW-GRASS has been discussed/presented in this test case. This issue will persist as long as authors do not expand the focus of their exercise. Authors also argue that "Most GIS tools will artificially fill these pits, but r.watershed's least-cost-path algorithm is able to route flow across them in a more realistic way". This is a quite interesting technical aspect to show with a practical example where you substantiate on relevant modeling issues using GSFLOW-GRASS. As it is right now the reader has no concrete results to get a better insight on what the authors state.

*To address the referee's comment that there was "nothing new" because we used the existing GRASS-GIS extension r.watershed: **our re-written Section 3.2.1** now explains that the r.watershed algorithm has never been used before with integrated hydrologic modeling, despite its superior performance compared to the local gradient-based tools used by others. This should now make clear that our introduction of the least-cost flow algorithm (r.watershed) to integrated hydrologic modeling is in fact an innovative step. The computational and performance specifications of the least-cost flow algorithm relative to other commonly used methods are now also fully documented in the rewritten Section 3.2.1 based on a thorough literature review, which addresses the referee's other request for more concrete aspects about the algorithm's advantages.*

The actual section on the Cannon River example was heavily edited to now point to Section 3.2.1.; reiterate the fact that other domain builders for integrated hydrologic models use different flow-routing algorithms that can be problematic; and emphasize that we created various new GRASS extensions in order to integrate r.watershed with GSFLOW (so that we are not simply plug-and-chugging with an existing tool):

> ***(p. 27 line 25-33)***
> ***"In such settings that lack integrated drainage, downslope flow-routing and "pit-filling" algorithms that are typically used to build hydrologic model domains***

(e.g., Bhatt et al. 2014, Maxwell et al. 2017, Gardner et al. 2018) can fail or produce spurious results by inappropriately modifying the real topography. As described in Section 3.2, GSFLOW-GRASS determines surface-water flow using the GRASS GIS's efficient and accurate r.watershed extension, which implements a least-cost path algorithm designed to produce drainage networks that route flow in the long-range path of steepest descent regardless of the degree of local drainage integration. By using r.watershed alongside a set of new GRASS-GIS extensions that integrate it into the GSFLOW framework, GSFLOW-GRASS is able to automatically create a topologically correct and linked drainage network in settings that lack integrated drainage for hydrologic model simulations."

(iv)What's the challenge associated to a large basin? Technically a small basin resolved at high resolution could be the same of a coarse-resolution large basin.

Indeed, a coarse-resolution grid of a large basin has the same computational complexity of a fine-resolution grid of a small basin. The question, then, is how to get to a reasonable coarse-resolution grid of a large basin - and this is what we address with Cannon River. We edited the text to clarify that as the largest of the examples watersheds, Cannon River takes advantage of the efficient irregular surface grid and a robust downscaling method of GSFLOW-GRASS:

(p. 27 line 16-20)
"Covering 3723 km^2, the Cannon River watershed is by far the largest of the three model implementations (Table 2) and benefits from the efficiency of the topographically based surface grid and the hydrologic robustness of the grid coarsening method in GSFLOW-GRASS. 17,455,046 flow-routing grid cells, each of which is 225 m^2 in area, were converted to only 610 irregular HRUs of >=10km^2 in area. For the groundwater domain, the elevation data were coarsened and hydrologically corrected to a 1 km regular MODFLOW grid."

Other points:

1. Captions still contain extensive discussion of the results that should be moved in the main text.

We wish to strike a balance between concise and informative captions and have maintained a moderate amount of the original text in the captions. However, we do recognize that some of the original caption text had not been adequately incorporated into the main text, and we now include it in the main text and do shorten some of the captions. ***Changes were made throughout each of the three examples to reference all the figures.***

2. I disagree with this statement:

"Unlike PIHM, GSFLOW-GRASS employs regular groundwater grid cells that are distinct from the irregular surface units, which makes the integrated domain building more complicated but allows for more complex representation of the surface-water and aquifer systems."

PIHM implements a fully unstructured grid that allows for a detailed representation of surface-subsurface water interactions. See reference below:

D. Wang, Y. Liu, and M. Kumar (2018). Using nested discretization for a detailed yet computationally efficient simulation of local hydrology in a distributed hydrologic model. Scientific Reports.

We appreciate the reviewer bringing this up and did not mean at all to imply that PIHM has fewer capabilities than GSFLOW. We only meant for this sentence to make clear that there are differences between the models, and that GSFLOW-GRASS' heterogeneous domain is more complicated to build but can be flexible. We have edited the sentence so that it no longer says "allows for more complex representation" and instead we now say that it "allows for flexible representation." The only form of complexity that could be easier represented with GSFLOW-GRASS than PIHM is vertical aquifer heterogeneity, since the MODFLOW component of GSFLOW-GRASS can have multiple layers whereas PIHM currently has a single vertically integrated saturated zone layer. However, we have no interest in comparing models in this manuscript - we in fact recognize that PIHM has a number of strengths relative to GSFLOW-GRASS. The full modified sentence is:

> ***(p. 24 line 14-p. 25 line 2)***
> ***"Unlike PIHM, GSFLOW-GRASS employs a regular three-dimensional groundwater grid that does not align with the irregular surface domain; this makes the integrated domain building more complicated but allows for a flexible representation of the surface-water and aquifer systems."***

We also thank the referee for sharing the new PIHM reference - we had not yet seen it. We added a citation to it earlier in the paper, where we discuss PIHM:

> ***(p. 4 line 15-17)***
> ***"In the case of PIHM (Qu and Duffy 2007), TINs were also implemented for better water balance performance through the mass-conserving finite volume method (Leveque 2002); further, nested TINs can provide efficient solutions when higher resolution is desired for certain target areas (Wang et al. 2018)."***

~~GSFLOW-GRASS~~ 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.

The 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~~GSFLOW–GRASS, a bundled set of open-source tools that develops inputs for, executes, and graphically displays the results of GSFLOW, the U.S. Geological Survey's coupled groundwater and surface-water flow model. In order to create a robust tool that can be widely implemented over diverse hydro(geo)logic settings, we built a series of GRASS GIS extensions that automatically discretizes a topological surface-water flow network that is linked with the underlying gridded groundwater domain. As inputs, ~~GSFLOW-GRASS~~ GSFLOW–GRASS requires at a minimum a digital elevation model, a precipitation and temperature record, and estimates of channel parameters and hydraulic conductivity. We demonstrate the broad applicability of the toolbox by successfully testing it in environments with varying degrees of drainage integration, landscape relief, and grid resolution, as well as the presence of irregular coastal boundaries. These examples also show how ~~GSFLOW-GRASS~~ GSFLOW–GRASS can be implemented to examine the role of groundwater–surface-water interactions in a diverse range of water resources and land management applications.

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. The widespread use of integrated models is stymied, however, by labor-intensive requirements for creating consistent sets of extensive model inputs, including the challenges of generating computationally robust surface and sub-surface model domains.

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 previously 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 to applications outside of the US (e.g., Hassan et al., 2014; Tian et al., 2015).

The process of implementing GSFLOW includes many hurdles that require significant time and computational knowledge to overcome. GSFLOW is not “fully integrated" in the sense that it does not simultaneously solve surface and subsurface flow equations; instead it consists of an iterative coupling between MODFLOW and PRMS that requires nearly all the individual input files for each of the two original models as well as an additional GSFLOW-specific linkage file. While a fully integrated model may have all the input information streamlined into a small number of internally consistent and efficiently organized files, to run GSFLOW, the user bears the burden of generating a multitude of diversely formatted ASCII files and ensuring that they contain inputs that correctly correspond with each other and can produce convergent coupled simulations. Freely available USGS GUIs – ModelMuse (Winston, 2009) and the PRMS GUI (Markstrom et al., 2015) – and proprietary GUIs (mostly for MODFLOW) can help users separately develop inputs to the two individual base models but do not offer support for creating the GSFLOW linkage file. The company Earthfx (http://www.earthfx.com/) provides full GSFLOW support as part of their “VIEWLOG” package, designed for the environmental consulting industry. More openly accessible software endeavors have also improved the usability of integrated hydrologic models ~~(Bhatt et al., 2014; Tian et al., 2016; Gardner et al., 0)~~ (Bhatt et al., 2014; Tian et al., 2016; Gardner et al., 2018) , but the community still lacks a free and complete package spanning pre- to post-processing for heterogeneous surface and subsurface domains. This lack of support for developing integrated hydrologic models such as GSFLOW motivates our present work, which we anticipate will enable more widespread hydrologic modeling.

Our overarching goal is to develop a bundled package – “~~GSFLOW-GRASS~~GSFLOW–GRASS” – to handle the complexity of the coupled GSFLOW model, thus tackling the grand challenge of accessibility plaguing many integrated modeling systems.

We develop an integrated toolbox featuring fully automated, robust, and open-source codes that cover the entire model implementation process within a consistent and efficient framework, from building topologically linked hydrologic domains and assembling model input parameters to visualizing model outputs. Our use of only free and open-source programming languages and software is a key feature of the toolbox's accessibility. Python scripts generate model input files and model output graphics, and extensions using the open-source GRASS GIS platform build topographically defined sub-watersheds linked to subsurface grid cells. Open-source software facilitates implementation of ~~GSFLOW-GRASS~~ GSFLOW–GRASS by diverse academic, government, and individual entities, enables further community development of ~~GSFLOW-GRASS~~GSFLOW–GRASS, and aligns with the USGS's goal to make its resources publicly accessible.

Developing a fully automated toolbox that can be readily executed for diverse physical settings raises the key technical obstacle of how to robustly build stream networks and sub-basins linked to subsurface computational domains without labor-intensive user intervention. Whereas overland flow routing and the calculation of drainage basins from topography are standard GIS capabilities, our tool improves upon these by automatically building topologically structured vectorized drainage networks without manual corrections using a least-cost path approach (Metz et al., 2011), while also including information on adjacency and routing pathways through the network that is required by integrated hydrologic models. The main technical advancement of ~~GSFLOW-GRASS~~ GSFLOW–GRASS is the development of streamlined GRASS GIS extensions that have passed a diverse range of stress tests, including steep to low-relief topographies, large and intricate to small and simple drainage systems, incomplete to full topographic drainage integration, and ~~mountainous~~ inland to coastal watersheds. These new capabilities enable rapid, automated delineation of surface-water drainage networks linked to subsurface domains across any generalized landscape and computationally feasible resolution within the range of scales permissible by GSFLOW. By doing this all within a framework that also includes open-source model input and post-processing tools, ~~GSFLOW-GRASS~~ GSFLOW–GRASS presents a solution toward more accessible integrated hydrologic modeling.

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. It is designed for simulations of watersheds with areas of a few square kilometers to several thousand square kilometers (Markstrom et al., 2008, p. 2). Although GSFLOW can run in modes equivalent to the stand-alone PRMS-IV model and the stand-alone MODFLOW model, only the "integrated" 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 the MODFLOW sub-component. 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) document the full details of the model.

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) of any shape as its fundamental discretized unit (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). Model intercomparison projects have included both representatives that use gridded domains and those that use irregular domains (Reed et al., 2004; Maxwell et al., 2014). In general, gridded domains are easier to construct and extend readily to parallelized computational systems, and they allow flexible spatial specification of soil and land-cover heterogeneity. In contrast, ungridded domains, such as triangulated irregular networks (TINs) used in models including tRIBS (Vivoni et al., 2004) and PIHM (Qu and Duffy, 2007), can conform more efficiently to complex terrain. In the case of PIHM (Qu and Duffy, 2007), TINs were also implemented for better water balance performance through the mass-conserving finite volume method ~~(Leveque et al. 2002)~~(LeVeque, 2002) ; further, nested TINs can provide efficient solutions when higher resolution is desired for certain target areas (Wang et al., 2018) . 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).

Vertically, the PRMS sub-component of GSFLOW 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.

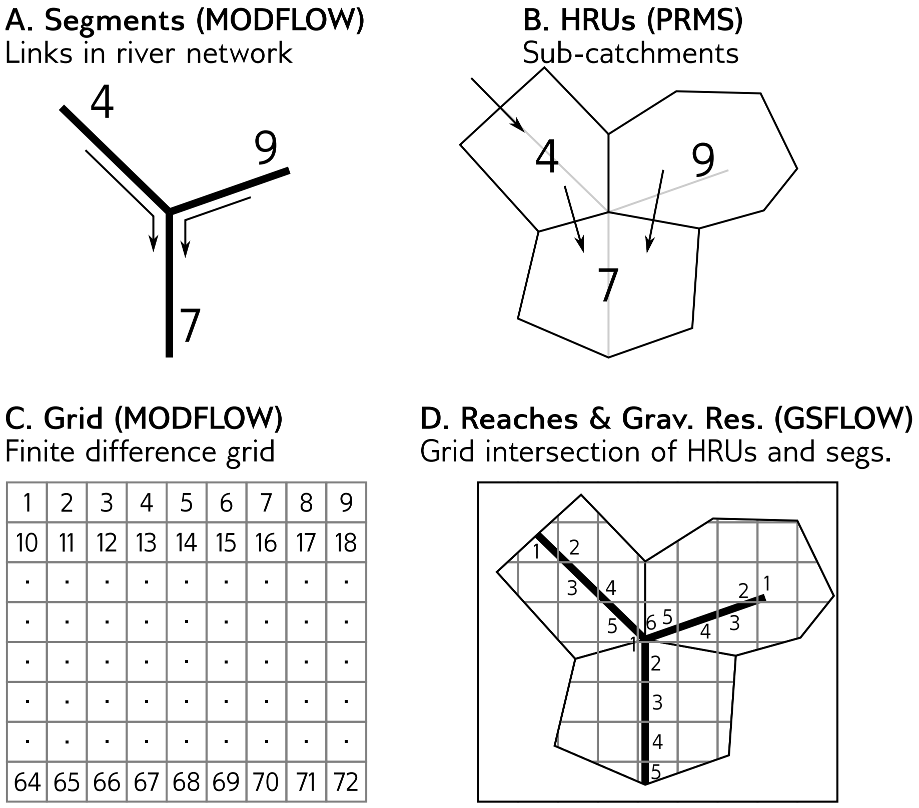

Figure 1. Major features of the GSFLOW geometry. **A.** Each segment is one link in the network. At each node, two tributary segments combine to flow into a single segment. Each is numbered. They need not be in any particular order, as indicated, but a downstream-increasing numbering scheme is required for updated inflows to all segments to be computed during the same iteration. **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 that they enclose. 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.

2.1.2 Process description

This section includes a brief description of the main hydrologic processes represented in GSFLOW, with select parameters listed in Table 1. Full details can be found in the GSFLOW manual (Markstrom et al., 2008). In particular, Table 1 from Markstrom et al. (2008) summarizes all the surface-water processes captured by PRMS modules, groundwater processes captured by MODFLOW stress packages, and model coupling procedures captured by GSFLOW.

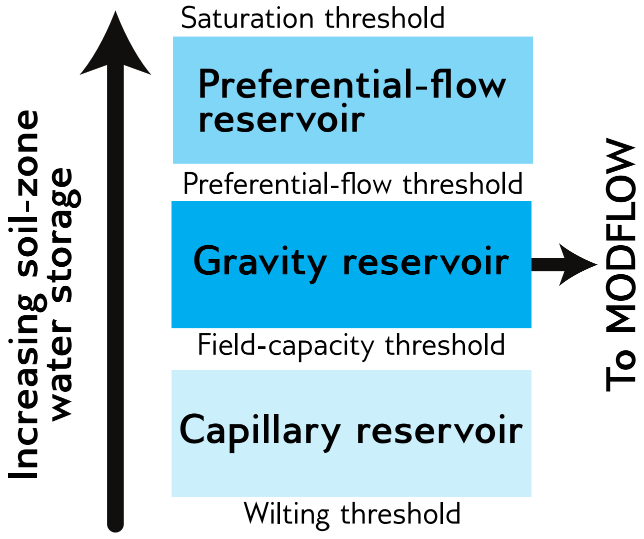

Figure 2. ~~(Adapted from Markstrom et al. (2008) , Figure 12) Soil water~~ Soil-water storage reservoirs in the PRMS component of GSFLOW. Within each HRU, ~~soil water~~ soil-water accounting calculations are carried out for three conceptual reservoirs in the order of increasing water storage and according 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 occur in the preferential-flow reservoir. (Adapted from Markstrom et al., 2008, Figure 12.)

.

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

For unsaturated zone flow, PRMS does not implement the Richards equation but instead applies computationally fast soil-water routing 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*) or drain vertically into the deeper subsurface domain that is handled by MODFLOW (based on parameters *ssr2gw_rate*, *ssr2gw_exp*, and *ssrmax_coef*). Gravity reservoirs can also receive groundwater discharge from the 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. Horto-

Table 1. Select GSFLOW parameters (adapted from Markstrom et al., 2008, Appendix 1).

Parameter	Description
pref_flow_den	Decimal fraction of the soil zone available for preferential flow versus capillary zone flow
soil_moist_max	Maximum available capillary water-holding capacity of soil zone
soil_rechr_max	Maximum quantity of water in the capillary reservoir (value must be less than or equal to *soil_moist_max*)
soil_type	Soil type: 1=sand; 2=loam; 3=clay
soil_moist_max	Maximum volume of water per unit area in the capillary reservoir
slowcoef_lin	Linear flow-routing coefficient for slow interflow
slowcoef_sq	Non-linear flow-routing coefficient for slow interflow
ssr2gw_rate	Linear coefficient in the equation used to compute gravity drainage to MODFLOW finite-difference cell
ssr2gw_exp	Exponent in the equation used to compute gravity drainage to MODFLOW finite-difference cell
ssrmax_coef	Maximum amount of gravity drainage to MODFLOW finite-difference cell
sat_threshold	Maximum volume of water per unit area in the soil zone, between field capacity and saturation thresholds
hru_percent_imperv	Decimal fraction of HRU area that is impervious
ICALC	An integer value used to indicate method used to calculate stream depth in this segment
IRTFLG	An integer value that flags whether transient streamflow routing is active

nian (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.

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 the finite difference method 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's law with specified streambed hydraulic properties. 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.

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 was significantly altered to facilitate its linkage with a MODFLOW subsurface domain. Other modifications are noted in the GSFLOW manual (Markstrom et al., 2008, 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 (Harbaugh, 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; úri 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 database management. ~~GSFLOW-GRASS~~ GSFLOW–GRASS utilities are written for the 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, which is critical for building stream networks; (3) its broad library of built-in hydrologic tools include those for vectorized drainage network development with downstream-increasing indexing (Jasiewicz and Metz, 2011) , which is essential for setting flow paths and adjacencies; (4) its generic Python scripting library and PyGRASS Application Programming Interface (API) make it easy to develop new extensions; (~~4~~5) 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~~6) it provides a GUI and command-line interface (CLI) that are consistent with one another. The GUI and CLI interfaces are not required for ~~GSFLOW-GRASS~~ GSFLOW–GRASS because the GRASS GIS component is handled mostly behind the scenes by a batch-processing Python script (**buildDomainGRASS.py**, Section 3.2); however, they allow end-users to re-run certain portions of the process and/or produce their own workflows using the ~~GSFLOW-GRASS~~ GSFLOW–GRASS extensions as building blocks. The open-source aspect of the present work is in part motivated by the need for water assessment and planning tools in the developing world (Pal et al., 2007), and these extensions, combined with the interchangeable and consistent GUI and CLI, can help users to generate their own advanced customizations of ~~GSFLOW-GRASS~~GSFLOW–GRASS.

3 Methods

We adopt a heterogeneous surface and subsurface computational domain for ~~GSFLOW-GRASS~~ GSFLOW–GRASS that employs sub-basin surface HRUs that are linked to subsurface grid cells. In addition to the computational efficiency of discretizing complex terrain ~~with~~ into sub-basins ~~rather than~~ with complex shapes rather than using a gridded surface domain at the resolution required to resolve the HRUs, the use of sub-basin HRUs that route surface runoff directly to stream segments also eliminates the need for establishing a cascading network (Section 2.1.2). Because of GSFLOW's conceptual ~~water-routing scheme (versus~~ (rather than gradient-based) surface-water-routing scheme, numerical differences between sub-basin and gridded HRU's are difficult to predict, but the automated ~~GSFLOW-GRASS~~ GSFLOW–GRASS toolbox can help enable future testing to rigorously interrogate their respective performances.

~~GSFLOW-GRASS~~ GSFLOW–GRASS strikes a balance between generating a ready-to-go GSFLOW implementation and providing flexibility to customize applications. With a newly developed set of automated and robust GIS domain builder tools, ~~GSFLOW-GRASS~~ GSFLOW–GRASS can be applied to any ~~DEM~~ digital elevation model (DEM) to produce GSFLOW model simulations~~, with only~~ . Only a few steps are required to set up ~~the model with~~ a GSFLOW model on the user's computer ~~directory~~ system. For further model-tuning, all scripts in the toolbox are open-source and commented to allow changes to any parameter ~~and~~ as well as development of optional GSFLOW capabilities not included in the default ~~GSFLOW-GRASS~~ GSFLOW–GRASS implementation. Many popular hydrologic model implementation programs have GUIs, including ModelMuse (Winston, 2009), Visual Modflow (Waterloo Hydrogeologic Inc., 2011), Hydrus (Simunek et al., 2009), ArcSWAT (Neitsch et al., 2002), and MIKE-SHE (Butts and Graham, 2005). While these are easiest for novice model users, GUIs can be challenging to develop for cross-platform implementations and generally support less flexibility for customization. Thus, we chose a mostly command-line approach, which has been designed and tested for use on Linux and Windows operating systems.

3.1 User-specified settings and model inputs

To seamlessly unify the different ~~GSFLOW-GRASS~~ GSFLOW–GRASS functionalities, including the automated GRASS GIS domain builder, GSFLOW input-file builder, and visualization components, users specify model inputs and configurations using a Settings text file. All inputs from the Settings file are read in and processed by the **ReadSettings.py** script. GSFLOW requires a daunting number of different model inputs (nearly 200 parameters for the PRMS sub-component alone). For ease of use, only a handful of application-specific and commonly adjusted inputs may be assigned using the Settings file, and default parameter values are applied elsewhere. While the default (and simplest) approach to ~~GSFLOW-GRASS~~ GSFLOW–GRASS is to modify only the Settings file, other parameters (including those mentioned in Section 2.1.2) may be readily changed in its input-file builder by searching for the parameter names defined in the GSFLOW manual and changing their values. The open-source nature of our toolbox also allows users to add parameters to the Settings files for future extensions of ~~GSFLOW-GRASS~~GSFLOW–GRASS.

Specifying and including spatially variable properties is a major challenge to distributed modeling. The Settings file accommodates the use of variable aquifer hydraulic conductivity, channel width, and Manning's n parameters, which are described

further in Section 3.3.3. Universal solutions are beyond the scope of the default toolbox, but we do provide a generalized GRASS-GIS extension called **v.gsflow.mapdata** to facilitate the generation of heterogeneous model inputs. **v.gsflow.mapdata**, further described in Section 3.2.4, can take any spatially variable data in a raster or vector GIS format and map it to one of the GSFLOW discretization structures: sub-basin HRUs for PRMS surface-water processes, regular grid cells for MODFLOW groundwater processes, gravity reservoirs that link the HRUs and MODFLOW grid cells, or stream segments or reaches for MODFLOW streamflow processes. This allows users to add data from any source – e.g., meteorological forcing, soil properties, hydrogeologic stratigraphy, or vegetation / land cover – to the ~~GSFLOW-GRASS~~ GSFLOW–GRASS data structures. Other software tools have facilitated hydrologic modeling by automating the connection with established databases ~~(Viger and Leavesley, 2007; Leonard and Duffy, 2013; Bhatt et al., 2014; Gardner et al., 0)~~ (Viger and Leavesley, 2007; Leonard and Duf
The USGS's GIS Weasel tool (Viger and Leavesley, 2007) may be used for deriving PRMS parameters from physical data sets such as STATSGO, which can then be mapped to the appropriate GSFLOW data structure using **v.gsflow.mapdata**. The current ~~GSFLOW-GRASS~~ GSFLOW–GRASS release aims to provide a general set of tools and does not directly link with any specific databases, which are typically only available in observation-rich regions and countries.

The Settings file is divided into subsections, each of which drives a portion of the model setup and organization. The "paths" section defines the computer directory structure for the project and GSFLOW executable, as well as the project name and GSFLOW version. Three GRASS GIS sections, "GRASS_core", "GRASS_drainage", and "GRASS_hyrdaulics", set the GIS location and path to the DEM, the surface and subsurface flow discretization parameters, and open-channel flow geometry and resistance, respectively. 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 a specified infiltration rate (see below) to calculate reasonable initial groundwater head conditions for the subsequent transient simulation that includes both the surface and subsurface domains; the steady-state step can be essential for obtaining numerically convergent groundwater results and more realistic solutions for the entire coupled system. 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" coupled runs without the preliminary groundwater steady-state period. The "time" section is used to specify the temporal window of the simulation. The "climate inputs" section sets input parameters for the PRMS "climate_hru" option, which is the standard climate implementation supported by ~~GSFLOW-GRASS~~ GSFLOW–GRASS (see Section 3.3.1) . Finally, the "hydrogeologic_inputs" section defines the preliminary steady-state MODFLOW infiltration rate, used for "spin-up" runs, and either a layered or fully distributed subsurface hydraulic conductivity structure. The **ReadSettings.py** script uses these inputs to create a directory structure and 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.

3.2 GRASS GIS domain builder

A critical challenge for any distributed hydrologic model is the fully automated development of a reproducible, topologically correct, and interlinked data structure that describes water flow through a catchment in a computationally efficient manner.

Semi-automated approaches to building surface flow networks are common (e.g., Luzio et al., 2006; Arnold et al., 2012), but the development of a fully automated approach has been impeded by the mathematical and logistical difficulties of building a topologically ideal drainage network (i.e. one whose fundamental unit is a tributary junction). Many standard GIS tools encounter problems when handling complex digital topography (represented using a DEM) that may contain natural or artificial depressions and whose grid cells are often much larger than real topographic features. Further complications arise when incorporating surface flow networks into integrated hydrologic models, because each link within the network must then be tagged with sufficient information to identify drainage pathways through the whole network, and the stream network must also be linked with ~~generally~~ sometimes different geometries and resolutions for surface-water ~~HRUs~~ and the groundwater-flow ~~grid~~grids.

We ~~have solved this general problem for any raster data set whose values (e.g., elevation) may be used to define a flow path, and we implemented our solution in a set of GRASS GIS extensions to generate flow networks for GSFLOW. This is done by generating a topologically correct drainage network whose base unit is the tributary junction, in which two stream segments meet and form a new segment. This simple set of rules, based on a least-cost-path drainage algorithm (Metz et al., 2011) , addresses systematic issues that may occur in other flow-routing algorithms and require users to manually perform error checks and corrections, which add a source of subjectivity and laborious processing time. While the automatically-generated stream and HRU networks will be topologically correct, their accuracy will be a function of digital elevation model (DEM) resolution, the topographic expression of the channel, and artificial drainage structures that may have minimal or no topographic expression. Therefore, they may also be edited by hand to match the geometry of features (such as artificial drainage structures) that are not included on the DEM. For each stream segment, the unique ID is recorded of the segment to which it sends its water, and this book-keeping is used to define surface-water flow through the network. The same ID number is assigned to the sub-basin HRU associated with the corresponding stream segment and its outlet. A MODFLOW grid is then built that is aligned with, but may be coarser than, the resolution of the DEM used for flow routing. Elevation values of the MODFLOW grid are then populated through a hydrologically correct downsampling of the DEM. From these fundamental surface-water and groundwater units, reaches and gravity reservoirs are generated based on the intersection of each segment and HRU, respectively, with the underlying MODFLOW grid. Unique identifiers are then passed between all of these in order to build a fully linked surface and subsurface flow network.~~

~~We created~~ addressed this challenge by creating eleven new GRASS GIS “extensions”, also called “add-ons”, that work alongside core GRASS GIS commands to transform user inputs (including a single DEM) into a set of GSFLOW inputs~~via the procedure outlined above and described in greater detail below. The GSFLOW 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 only one time using GRASS GIS, and then perform multiple runs for parameter calibration or scenario tests without having to repeat the domain construction. This~~ . This workflow creates an automatically generated network of streams and HRUs that is spatially linked to a MODFLOW grid. The domain-building procedure is automated through the **buildDomainGRASS.py** script, which takes inputs from the Settings text file, implements the domain-building workflow, and produces ASCII files used by ~~GSFLOW-GRASS~~GSFLOW–GRASS's Python input-builder scripts.

3.2.1 Surface-water network

In the first step of the fully automated domain-building workflow, GRASS GIS imports a user-provided DEM to define the drainage network and HRUs. After hydrologically correcting the DEM by filling pits and removing cells that have flow inputs from outside the map area (~~GSFLOW-GRASS~~ GSFLOW–GRASS requires the full topographical catchment to be included in the model domain), a Hortonian drainage network is constructed ~~(Jasiewicz and Metz, 2011; Metz et al., 2011)~~ using the **r/v.stream.*** toolkit (Jasiewicz and Metz, 2011) that relies on a single-flow-direction implementation of the **r.watershed** flow-routing algorithm (Metz et al., 2011) . Sub-basins associated with each stream segment are designated as HRUs in order to follow both the natural discretization of the landscape and the architecture of PRMS (Markstrom et al., 2015). River headwaters are defined based on a threshold drainage area that may be weighted by the user to represent, for example, nonuniform precipitation or snowmelt inputs. Such weights permit a more realistic representation of drainage density and, as a result, increased model resolution in areas that contribute more water to the catchment.

The GRASS GIS drainage-network-creation algorithm, **r.watershed** (Metz et al., 2011) , is both efficient and accurate. For computations that can take place entirely within memory, its speed exceeds that of both Terraflow and the D8 routing used by ArcGIS (Jenson and Domingue, 1988; Maidment and Morehouse, 2002; Arge et al., 2003; Magalhães et al., 2014) . This speed results from its sorting algorithm and priority queue, and a standard desktop workstation today can process DEMs in memory with tens of thousands of cells on each side. The least-cost path approach taken by **r.watershed** does not require any pit-filling step, but we do so in order to create a more consistent DEM with downslope-routed flow for the remainder of the analysis. The flow-routing component of the more recent "Fastscape" algorithm by Braun and Willett (2013) could be faster than **r.watershed**, but these have not been benchmarked, and Fastscape is not yet integrated into the GRASS GIS toolchain, which is necessary for all of the subsequent steps. Kinner et al. (2005) demonstrated that **r.watershed** is more accurate than Terraflow (Arge et al., 2003) , especially in low-relief areas and those in which tree canopy elevations are mistakenly interpreted as ground-surface elevations; this latter issue is pervasive across many digital elevation models, including the widely used Shuttle Radar Topography Mission (SRTM) DEM (Farr et al., 2007; Miliaresis and Delikaraoglou, 2009) .

In spite of these advantages, **r.watershed** has not before been used to build flow networks for integrated hydrologic models. Other integrated hydrologic model domain-building tools use local drainage direction information (Bhatt et al., 2014; Maxwell et al., 2017; While not an integrated hydrologic model due to its limited subsurface modeling capabilities, Srinivasan and Arnold (1994) integrated GRASS GIS version 4, including an earlier and much slower version of **r.watershed**, into the Soil & Water Assessment Tool (SWAT). Beyond this, **r.watershed** is typically discussed in the drainage algorithm literature (e.g., Barnes et al., 2014; Magalhães et al., 201 directly applied to flow-routing and cost-path calculations (e.g., Wickert et al., 2013; Bird et al., 2016) , or included as a component of an assessment tool (e.g., Bhowmik et al., 2015; Rossi and Reichenbach, 2016) . By integrating **r.watershed** into GSFLOW–GRASS, via the **r/v.stream.*** toolkit for Hortonion drainage network analysis (Jasiewicz and Metz, 2011) , we are able to harness the capabilities and efficiency of the hydrologic computational engine within GRASS GIS for integrated hydrological modeling.

Following drainage network construction, the next step in the automated workflow 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~~ , which builds atop the upstream-to-downstream stream-segment and HRU indexing in the existing **r/v.stream.*** toolkit (Jasiewicz and Metz, 2011) . This index is a unique positive integer identifier ~~,~~ applied to each segment and its overlapping HRU, and is called a “category” in GRASS GIS. For each segment and overlapping HRU in the drainage network (Figure 1A,B), **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. This links the stream segments and HRUs in the watershed as a directed graph (e.g., Czuba and Foufoula-Georgiou, 2014; Heckmann et al., 2014; Tejedor

At this point, the user may ~~make edits to the structure of the drainage basin, for example, by correcting stream courses to align with~~ optionally break out of the automated workflow and edit the vector geometries that define the streams and sub-basins. While we expect that many users will find the fully automated approach to be a major advantage over those that require manual intervention – these add a source of subjectivity and laborious processing time – Gardner et al. (2018) note that human-developed drainage structures ~~.~~ may cause a discrepancy between topographically routed flow and actual flow paths. This manual adjustment is not standard, and requires the addition of a break point in **buildDomainGRASS.py**, as well as for the user to manually adjust the category numbers (indices) and “tostream” network topology values in the attribute tables for the segments and HRUs if the changes are substantial enough to change the flow network.

After this, the study area is limited to a single drainage basin using the new **v.stream.inbasin** extension, completing the development of the drainage network geometry and topology. This step is included because the goal of many hydrologic studies is to understand a single watershed basin. If this is not the case, **buildDomainGRASS.py** may be edited to skip this step and to analyze all complete drainage networks within the domain.

Each stream segment is then supplied with ~~attributes~~ attribute values required for GSFLOW through the **v.gsflow.segments** extension. This numbers each ~~river channel~~ segment for GSFLOW (Figure 1A) and populates the associated database table with hydraulic geometry, channel roughness (constant or spatially distributed), and channel and floodplain width (constant or spatially distributed). Additional less-commonly used options are also available, including additional input discharge for the upstream-most stream segments (e.g., from human intervention), input diffuse runoff, and direct precipitation on the stream.

~~The~~

3.2.2 Groundwater-flow grid

Following the completion of the surface-water domain, the next step is to build the ~~MODFLOW grid and link it to the surface-water data structures (HRUs and segments). The primary difficulty is that the MODFLOW grid cells can be an arbitrary size and may not overlap with the irregularly shaped HRUs and segments. Furthermore, it~~ groundwater domain. MODFLOW-NWT uses a rectangular finite-difference grid structure (Harbaugh, 2005; Niswonger et al., 2011) . The cell size for this grid is selected by the user in the Settings file. It is often desirable to discretize the MODFLOW groundwater domain on a grid that is coarser than the DEM used for surface flow routing ~~, for the sake of computational efficiency~~ in order to increase computational efficiency while still allowing GSFLOW–GRASS to generate a complex surface-water network; the proper grid cell size depends on the size of the HRUs and the strength of the surface-water–groundwater coupling. **v.gsflow.grid** builds the MODFLOW grid (Figure 1C) using the built-in **v.mkgrid** command while enforcing that it must contain only whole DEM grid cells, and that its edges must align with cell edges in the DEM.

Following grid creation, which often includes coarsening of the original DEM, **r.gsflow.hydrodem** then hydrologically corrects the elevations of the MODFLOW grid cells~~; cells~~. Cells that contain stream segments are given a surface elevation corresponding to the lowest-elevation overlapping pixel on the fine-scale flow-routing DEM, while all other MODFLOW cells are assigned the mean elevation from the corresponding cells in the flow-routing DEM. This is crucial where a river valley is less than two grid cells wide: in this case, it is possible that the cell corresponding to the flow path will average over both the valley wall and the valley bottom, creating a bumpy valley floor that contains artificial dams. Thus, both the flow-routing DEM (high resolution) and the MODFLOW grid (typically, though not necessarily, lower resolution) are hydrologically corrected to enforce decreasing elevations down the drainage network.

3.2.3 Surface-water–groundwater coupling

The final step in developing the GSFLOW domain is to link the surface-water geospatial data structures (HRUs and segments) with the MODFLOW rectangular grid. **v.gsflow.reaches** and **v.gsflow.gravres** construct the reaches and gravity reservoirs (Section 3.1), which are the intersection of segments and HRUs, respectively, with each MODFLOW grid cell (Figure 1D). The database table for the reaches includes values for the thickness of the stream-bed sediment (defaults to 1 m) and its hydraulic conductivity (defaults to 5 m/d, characteristic of sand and gravel).

3.2.4 Accessing additional GSFLOW functionality

GSFLOW supports more input options than we have defined for our GRASS GIS **v.gsflow.*** commands, ~~but~~ though we have included many of the most common options~~, as well as~~. These are sufficient to set up and run a GSFLOW simulation. However, they may not encompass all of the variables that some users may consider to be important.

Therefore, GSFLOW–GRASS includes the **v.gsflow.mapdata** tool for users to add other attributes to database tables, with a focus on spatial distributions. These attributes can include spatially variable precipitation and temperature, parameter choices for model spin-up, and fully distributed maps of hydraulic conductivity, specific yield, streambed hydraulic properties, soil texture, vegetation type, and evapotranspiration parameters. The core capability of **v.gsflow.mapdata** is the use of averaging and nearest-neighbor methods to connect input data from raster grids, vector areas (polygons), or vector points, to the attribute tables of the HRUs, segments, gravity reservoirs, reaches, and/or MODFLOW grid cells. As these are custom additions, calls to **v.gsflow.mapdata** must be added by end users to **buildDomainGRASS.py**. Once added, the end user can follow our template code in the input-file builder to add these to the GSFLOW input files. **v.gsflow.mapdata** therefore adds user-driven flexibility which input data can be supported by GSFLOW–GRASS, and a starting point for users who may want to expand on its capabilities.

~~Finally~~

3.2.5 Geospatial data export

In the final step, the generated attributes and geometries are exported. This information is stored in GRASS GIS as raster grids and vector geometries associated with SQL database tables. **buildDomainGRASS.py** exports a rasterized "basin mask" (1 in the basin, 0 outside) and the hydrologically corrected DEM at the MODFLOW grid resolution, as well as vectorized GIS data (shapefile format) for the HRUs, gravity reservoirs, MODFLOW grid, stream segments, stream reaches, pour point, full study basin area, and downstream boundary-condition cells. **v.gsflow.output** exports the database tables associated with the vectorized GIS data in comma-separated variables (CSV) files that can be read in by the input-file builder scripts (Section 3.3) for use in GSFLOW. These exported data are then ready to be parsed into GSFLOW inputs using the Python input-file builder scripts (Section 3.3) and/or to be used for data visualization (Section 3.5).

This separation between the GIS and ASCII-input-file components is intentional. The GRASS domain-builder component typically requires several minutes to run, and often only needs to be executed once for a watershed. The ASCII files, on the other hand, can form an effective basis for ensembles of runs. These can be used to calibrate parameters or explore hydrologic sensitivity to variable forcing scenarios.

3.3 GSFLOW Input File Builder

~~GSFLOW-GRASS~~ GSFLOW–GRASS includes a set of input-file builder scripts that are streamlined to incorporate the model domain discretization constructed by the GRASS GIS workflow and generate corresponding model inputs for the GSFLOW control file, PRMS-type input files, and MODFLOW-type input files. Most of the new features in GSFLOW that are 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 use of a "control file" as the main interface file, "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. The following builder scripts handle these different formats and are automatically executed through the toolkit's Run file (Section 3.4). The builder scripts may also 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~~ 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; spatially variable entries can be generated with the aid of the **v.gsflow.mapdata** tool. The first two sections are required and

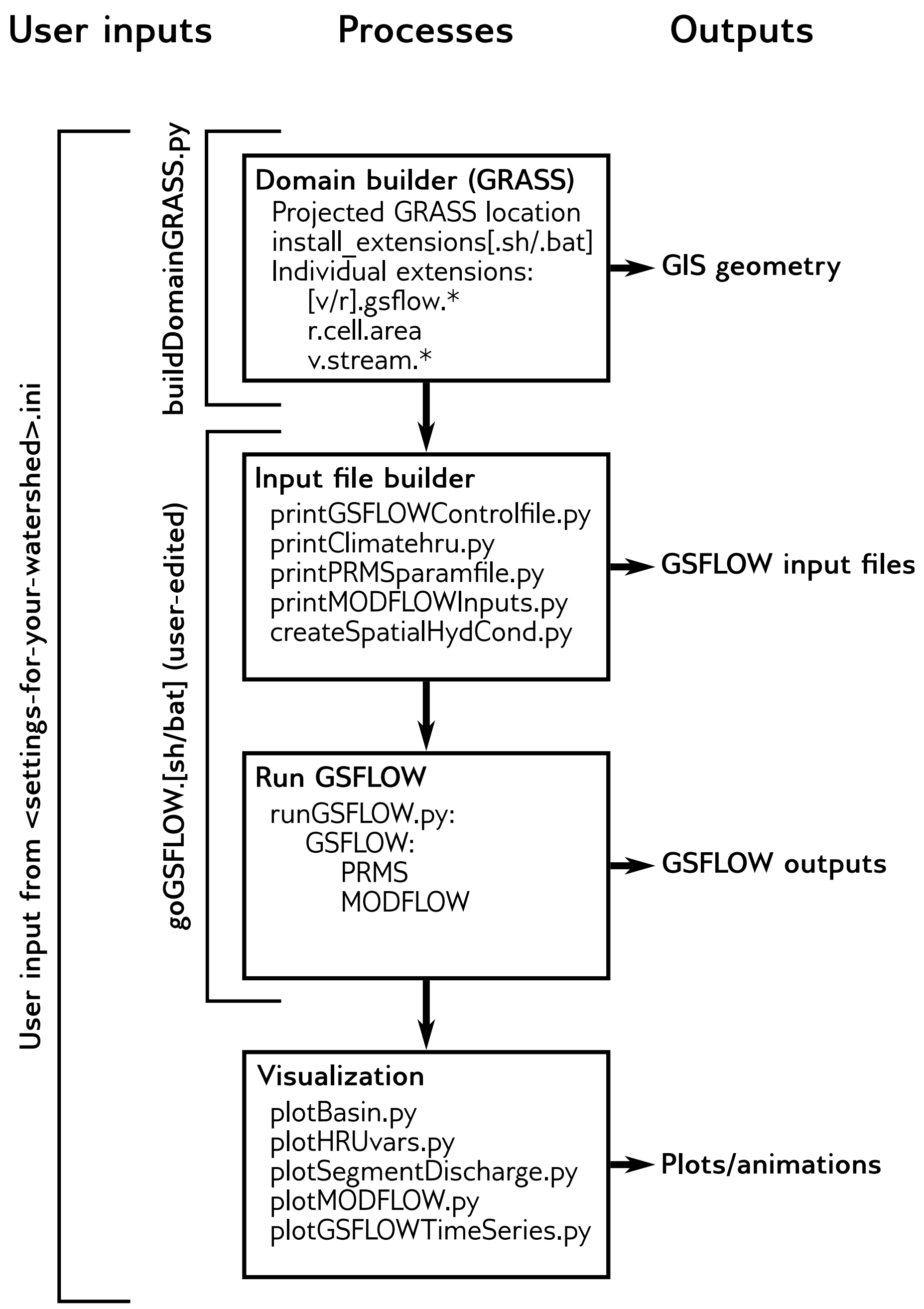

Figure 3. ~~GSFLOW-GRASS~~ GSFLOW–GRASS workflow. The user: (1) creates a ***.ini** file based on their study catchment; (2) creates a projected GRASS GIS location; (3) runs **buildDomainGRASS.py**; (4) edits and runs **goGSFLOW.py**. After this, they may use ~~GSFLOW-GRASS~~GSFLOW–GRASS's visualization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

.

include details about the simulation execution and module choices. The third section establishes spin-up versus restart run modes based on Settings file entires. Sections 4 and 5 contain customizable lists of output variables to be printed, which can be used by visualization scripts in ~~GSFLOW-GRASS~~ GSFLOW–GRASS (Section 3.5). 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 “climate_hru” module provides flexibility for the user to implement their own spatial interpolation or extrapolation methods, which can then be transferred to the GSFLOW domain with the **v.gsflow.mapdata** tool. ~~GSFLOW-GRASS~~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 or batch file for Windows) for running GSFLOW with the control file. After all other input files are created, this executable is 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~~ 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. For a quick set-up of ~~GSFLOW-GRASS~~GSFLOW–GRASS, the script **printClimatehru.py** 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. A spatially uniform approach may be acceptable where the size of a rainstorm is typically 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; these require custom inputs from the end-user, which can be ported from any discretization to the HRU domain with the aid of the **v.gsflow.mapdata** tool.

The parameter file is created by the script **printPRMSparamfile.py**. The script includes sections for domain dimensions and for parameters inputs, both of which are streamlined to 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, the parameter file requires a substantial number of parameter 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. Spatially variable information can be transferred to the HRU domain using the **v.gsflow.mapdata** tool. 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 the **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 a directory structure that is consistent with the rest of the input files and the visualization scripts.

The ~~GSFLOW-GRASS~~ 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 (e.g., *finf* entry in the Settings file). 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). For these, we recommend that users import the appropriate data sources into GRASS GIS and use the **v.gsflow.mapdata** extension to map these parameters onto the appropriate GSFLOW objects (e.g., HRUs, MODFLOW cells). 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 extensions. By default, the streamflow calculation is set to use Manning's equation by assuming a wide rectangular channel ($ICALC = 1$). Spatially variable stream widths and/or Manning's n values may be set through the Settings file, based on either gridded or point-based (e.g., survey) data, and **v.gsflow.segments** also supports the delineation of both channel and floodplain geometries and roughness parameters.

3.4 GSFLOW run file

For the user's convenience, the ~~GSFLOW-GRASS~~ GSFLOW–GRASS toolkit includes an executable Run file, which is a shell script for Linux, **goGSFLOW.sh**, and a batch file for Windows, **goGSFLOW.bat**. The Run file collects input from a specified Settings file and then runs all of the above input-file builder scripts; the script **runGSFLOW.py**, which launches the GSFLOW simulation; and the runtime visualization script **plotGSFLOWTimeSeries_Runtime.py**, further described below. If the runtime visualization is not desired, the user can comment out the corresponding execution line in the Run file. As long as the user does not wish to use more features than are exposed in the Settings file, no direct interface with the code is required to run ~~GSFLOW-GRASS~~GSFLOW–GRASS. This permits a "quick-start" implementations of GSFLOW, which can substantially lower the barrier to entry for using this model.

The Run file may be implemented only after the model domain is generated through **buildDomainGRASS.py**. The ~~GSFLOW-GRASS~~ GSFLOW–GRASS toolkit separates the GRASS domain-builder module from the Run file because users will typically only need to construct their domain once, but will perform multiple runs of the model with variable parameter inputs, for example, for model calibration or to simulate different time periods.

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 rerunning **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: QGIS Development Team, 2013) outside of ~~GSFLOW-GRASS~~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. Our toolbox also includes the runtime visualization script **plotGSFLOWTimeSeries_Runtime.py** that is by default called by the Run file (but can be commented out if desired) and displays a continuously updated time series plot of basin-total precipitation and discharge. Tracking simulation progress with runtime plots can be very useful for complex integrated models, which can have lengthy simulation times.

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 bodies 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 a 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~~ GSFLOW–GRASS is 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

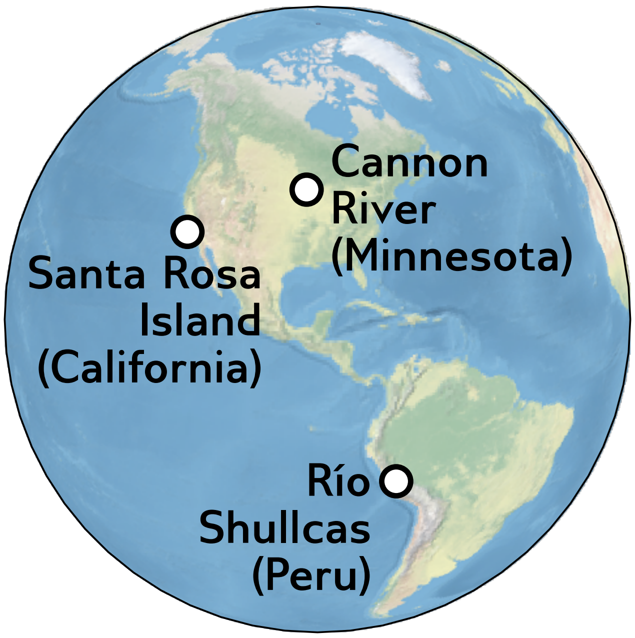

Figure 4. Our test sites include the high Andes, a mountainous island, and a formerly glaciated Mississippi tributary.

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

Three example implementations demonstrate (1) the variety of hydrological processes and environments that can be explored using ~~GSFLOW-GRASS~~GSFLOW–GRASS, and (2) how the toolkit's GIS domain builder can handle diverse topographic settings, including those prone to problems with standard GIS stream network tools. Towards ~~this~~ the first point, the specific examples chosen represent a range of practical applications for water and land management. Towards the second point, each simulation presents a unique set of technical challenges in developing a topographically based model domain that can properly route rainfall through a network of stream segments and sub-basins as well as a connected groundwater-flow grid. It is important to note, however, that no calibration effort was made to match field observations for these test cases. The simulation results thus serve as purely schematic examples based on certain settings and do not aim to capture actual conditions at the corresponding sites.

The examples are based on 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, USA, which has undergone land-cover change impacts; and the formerly glaciated Cannon River watershed, in which water flows from intensely farmed uplands into an incised bedrock valley in Minnesota, USA (Figure 4; Tables 2 and 3). 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 with ~~partially- to fully-integrated~~ partially to fully integrated drainage. Their catchment areas range from 12.5 km^2 to 3723.0 km^2, covering the range of scales that GSFLOW was developed to simulate. They are affected 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~~ grazing-induced erosion (Santa Rosa) (Schumann et al., 2016), and agricultural runoff and fertilizers (Cannon River) (Kreiling and Houser, 2016). 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, where financial limitations faced by local environmental researchers and practitioners make it difficult to use commercial software solutions.

Figures 5–7 display sample inputs and outputs of the model simulations using the default ~~GSFLOW-GRASS~~ GSFLOW–GRASS toolkit for the three test cases. These applications show that even before any parameter adjustments, the ~~GSFLOW-GRASS~~ GSFLOW–GRASS toolkit can readily generate GSFLOW model domains and parameter inputs that produce numerically convergent simulations in a variety of topographies and hydroclimatic conditions.

Preliminary simulations with the default ~~GSFLOW-GRASS~~ GSFLOW–GRASS provide a valuable springboard for the next step of performing the calibration needed to generate realistic model outputs for specific sites. The ~~GSFLOW-GRASS~~ GSFLOW–GRASS toolbox can be customized to quickly generate additional model runs with varying input values to expedite the parameter calibration. It can also facilitate the implementation of sensitivity or other Monte Carlo-type analyses that are critical for identifying issues such as equifinality and over-parameterization and for determining uncertainty estimates (Beven, 2006; Gallagher and Doherty, 2007; Razavi and Gupta, 2015; Song et al., 2015).

4.1 Shullcas River watershed, Peru

The first test case is based on 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 Glacier, which supplies meltwater to the Shullcas River, is rapidly retreating (López-Moreno et al., 2014), causing concern over future water resources (Somers et al., 2018) . However, in glacierized watersheds of the Peruvian Andes, a large proportion of the dry season stream discharge can be composed of groundwater (Baraer et al., 2015), 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.~~

~~A major obstacle with the steep topography and narrow canyons of Shullcas is the need for impractically high resolution and computational expense if using a standard gridded model domain. GSFLOW-GRASS's use of topographically based irregular~~

Table 2. Catchment and hydrological characteristics of the ~~GSFLOW~~ GSFLOW–GRASS example sites~~and their tests for GSFLOW-GRASS~~.

Site	Drainage area [km^2]	~~Elevation range~~ Flow-routing cellsize [m^2]	MODFLOW cellsize [km^2]	Min. HRU area [km^2]	Elevation range [m]	Mean annual rainfall [mm]	Daily rainfall CV ~~Tests of GSFLOW-GRASS capabilities~~
Shullcas River, Junín Region, Peru	161.4	~~3526–5527~~ 930.93	0.25	~~1.0~~ 1	3526–5527	1076	1.4 ~~Steep topography, seasonal rainfall~~
Water Canyon, Santa Rosa Island, California, USA	12.5	~~23–378~~ 8100	0.0324	0.4	23–378	265	5.4 ~~NULL cells defining irregular coastline, small basin, small number of coarse-resolution cells~~
Cannon River, Minnesota, USA	~~3723.0~~ 3723	~~203–413~~ 225	~~1.0~~ 1	~~10.0~~ 10	203–413	756	3.2 ~~Two-layer hydraulic conductivity, poorly integrated drainage, low relief, large basin~~

Precipitation statistics from 2013-08-26 to 2016-09-29 (Shullcas); 1990-04-23 to 2017-09-27 (Water Canyon); 1938-05-12 to 1943-11-05 (Cannon). "Flow-routing cellsize" is the resolution used to construct the stream network and irregular HRU cells, which are ultimately coarser-sized ("Min. HRU area"). CV = coefficient of variation.

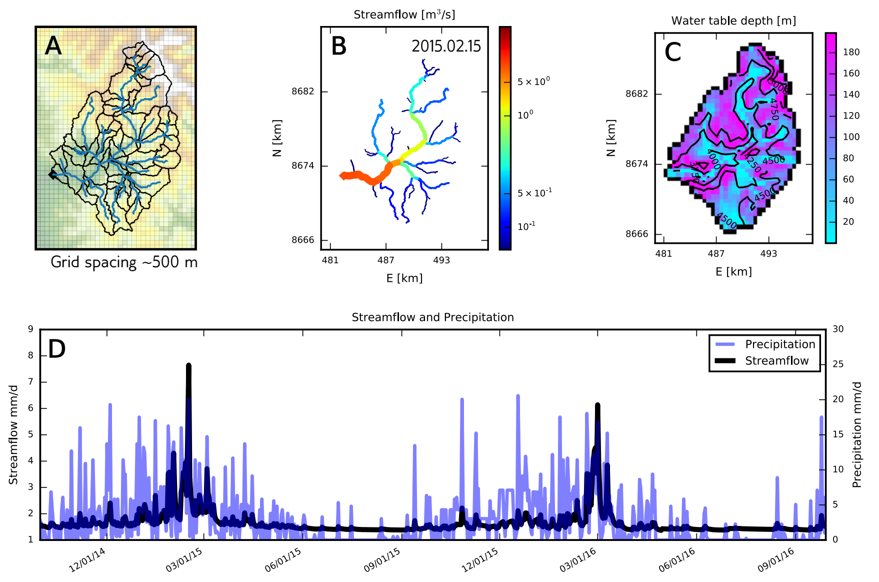

Figure 5. Model based on Río Shullcas, Peru. **(A)** Map with MODFLOW grid, HRU outlines, stream segments (blue), and digital elevation model. **(B)** ~~The model shows streamflow accumulating~~ Streamflow accumulation through the mountainous drainage network. **(C)** ~~Simuated groundwater levels are shallowest in low and flat areas~~The modeled water table distribution with elevation contours (m.a.s.l.). **(D)** ~~Both~~ Seasonally variable precipitation and streamflow~~are seasonal~~.

~~HRUs makes it possible to compute flow paths using high-resolution topography, and then convert these into a relatively small number of computational cells corresponding to fundamental surface-water hydrologic units – stream segments and sub-basins. In contrast, a surface discretization not automatically aligned with flow direction would require a greater number of cells to accurately resolve flow paths. The irregular drainage units generated by GSFLOW-GRASS are then linked to a groundwater (MODFLOW) grid, whose regular finite-difference grid can also be coarse relative to the high-resolution DEM. Cell elevations for the MODFLOW grid are assigned based on our hydrologically corrected downscaling method (Section 3.2). This combination of network-based surface-water routing and hydrologically corrected grid cell elevations for groundwater solves the problem of spurious dams and lakes that arise when routing flow across a rectangular grid in which grid cell elevations are averages across steep gradients.~~

The simple hydrologic model based on the Shullcas watershed covers an area of 161.4 km^2 and ranges in elevation from 3626 to 5527 m above sea level (a.s.l.) (Table 2). Using the GRASS domain-builder, the watershed was divided into 59 sub-basin HRUs based on an ASTER nominal 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 of 485 m, by 33 columns, each with a width of 492 m. Meteorological data were obtained from the Peruvian Meteorological Office (SENMAHI) online database.

The steep topography makes the Shullcas River Watershed an apt testbed for examining the ability of GSFLOW–GRASS to represent surface-water–groundwater links in challenging terrain. The major obstacle with Shullcas' steep topography and narrow canyons is the need for high resolution to represent surface flow, which leads to an impractically high number of computational units and expense if using a standard gridded model domain. An irregular surface grid can provide a much more efficient discretization, but this then entails painstaking indexing to link it to the subsurface grid, which must be a regular rectangular grid for the MODFLOW component of GSFLOW. Also, as the most computationally expensive part of GSFLOW, practical MODFLOW implementation typically needs a coarser resolution grid than that used for resolving the stream network, but simple coarsening of DEMs with steep gradients (e.g., by using mean elevations from the higher-resolution DEM) can result in hydrologically incorrect groundwater flow directions. GSFLOW–GRASS addresses these problems by computing flow paths using high-resolution topography ($\sim 2\times10^5$ grid cells of 900-m^2 size) and converting these into a far smaller number of larger computational surface cells (79 HRUs that are $\gtrsim$1 km^2 in area) that convey the same fundamental surface-flow information (see Table 2); this efficiency is possible because the surface is discretized along topographically defined surface-water hydrologic units – stream segments and sub-basins (Section 3.2.1). To create a coarsened subsurface rectangular grid domain for the MODFLOW component, GSFLOW–GRASS' hydrological correction to enforce integrated subsurface drainage (Section 3.2.2) proved essential for preventing unrealistic results. Early model tests for Shullcas showed that simple grid coarsening using the mean value of the elevations from the higher-resolution grid could, for example, average elevations between flat valley floors and steep canyon walls. This caused cells containing the stream to be higher in elevations than the surrounding surface on the groundwater flow grid, leading to lateral flow out of these "dams" that formed as a numerical artifact of averaging. As the final step of the domain-building solution, GSFLOW–GRASS extensions seamlessly link the hydrologically corrected coarse-scale MODFLOW domain with the irregular surface HRUs (Section 3.2.3).

The 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 precipitation variability. In simulations, ~~peak discharge occurs~~ discharge varies seasonally in response to precipitation, with peak discharge occurring late in the wet season, after significant antecedent moisture has built up within the catchment, and essentially constant baseflow ~~supports~~ supporting low but reliable discharge throughout the dry season ~~. The GSFLOW-GRASS~~ (Figure 5D). The GSFLOW–GRASS post-processing visualization tools were ~~used to depict the spatial distribution of water table depths relative to~~ useful for depicting the accumulation of streamflow throughout the drainage network (Figure 5~~B-C) , as well as time series of watershed forcing and responses~~ B) and water table depths that were shallowest in low and flat areas (Figure 5~~D~~C). The model simulates ~~most rainfall infiltrating~~ that most rainfall infiltrates to recharge the aquifer, with relatively little overland flow. This result likely underestimates actual surface runoff, considering the significant erosive overland flow events have occurred in the recent past (Wagner et al., 2004). Nevertheless, preliminary results depict groundwater converging at the stream network that can give information about whether baseflow can sustain discharge at the catchment outlet during dry periods.

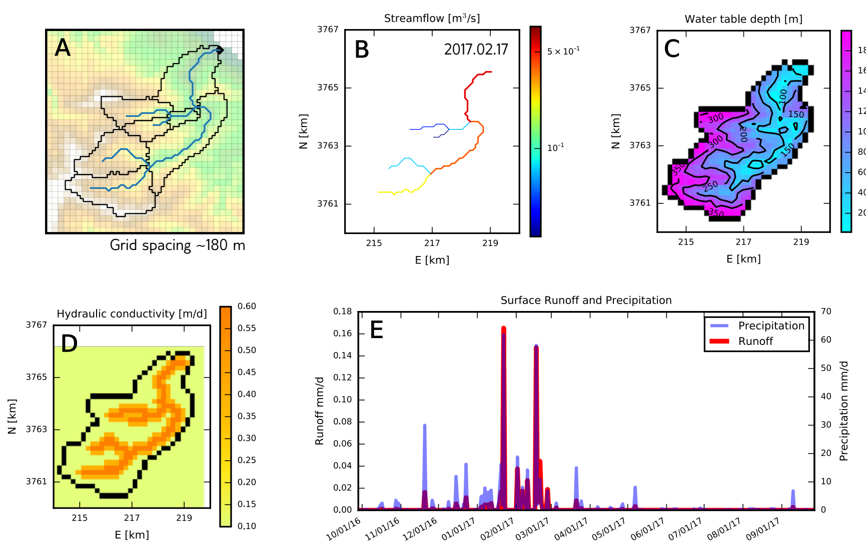

Figure 6. Model based on Water Canyon, Santa Rosa Island, California, USA. **(A)** Map with MODFLOW grid, HRU outlines, stream segments (blue), and digital elevation model. **(B)** Streamflow accumulation through the drainage network. **(C)** The modeled water table ~~is deepest below ridge tops and becomes shallow in the valleys~~distribution with elevation contours (m.~~(D) We generated~~ a~~spatially~~ .s.l.). **(D)** Spatially variable hydraulic conductivity structure, with hydraulic conductivity increasing near the channel to represent alluvium and colluvium. **(E)** Simulated surface runoff contributions to catchment-wide discharge ~~correlate in time~~ compared with precipitation~~, but are much lower in magnitude~~.

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 modeled for the 19 major drainages around the island during hypothetical climate regimes that are wet, dry, and of average wetness when compared to modern conditions. Unlike PIHM, ~~GSFLOW-GRASS employs regular groundwater grid cells that are distinct from~~ GSFLOW–GRASS employs a regular three-dimensional groundwater grid that does not align with the irregular surface ~~units, which~~ domain; this makes the integrated domain building more complicated but allows for ~~more complex~~ a flexible representation of the surface-water and aquifer systems.

Here we apply the ~~GSFLOW-GRASS~~ GSFLOW–GRASS toolbox to model Water Canyon (~~Table 2~~ Tables 2 and 3), one of the island's many drainages~~, and demonstrate its ability to generate small drainages covering just a few high-gradient DEM grid cells and an irregular but real-world boundary: the coastline. Water Canyon is unique among the three example sites~~

Table 3. Model implementations based on three sites serve to test GSFLOW–GRASS capabilities and demonstrate applications.

Site	GSFLOW–GRASS capabilities	Applications
Shullcas River, Junín Region, Peru	Efficient discretization of steep topography; Hydrologically corrected coarsening	Water resources in mountain catchments; Groundwater-surface water interactions under seasonally variable precipitation
Water Canyon, Santa Rosa Island, California, USA	Irregular / coastline boundaries; Hydrologically corrected coarsening; Spatially distributed hydraulic conductivity	Management of eroding hill slopes; Semi-arid climate with losing streams
Cannon River, Minnesota, USA	Two-layer hydraulic conductivity; Least-cost flow algorithm for poorly integrated drainage	Mixed agricultural—recreational watersheds; Strong temperature seasonality

Precipitation statistics from 2013-08-26 to 2016-09-29 (Shullcas); 1990-04-23 to 2017-09-27 (Water Canyon); 1938-05-12 to 1943-11-05 (Cannon). CV = coefficient of variation.

~~in that its outflow drains to the ocean, which is represented by cells with NULL values. This required modifications to the GSFLOW-GRASS source code in order to properly accumulate flow across the watershed without encountering errors due to the null values offshore. We drove this hydrologic model~~. We generated the surface flow routing system with topography derived from a 3 arcsecond SRTM DEM (Farr et al., 2007) projected to a UTM coordinate system at 90 m resolution~~. This resolution is already low compared to the size of the island drainages, but we further~~, and we down-sampled the DEM to 180 m resolution for the MODFLOW grid~~to test the performance of our toolbox with a coarse-resolution representation of a steep catchment.~~

~~GSFLOW-GRASS successfully produced simulation results~~. We drove simulations shown in Figure 6 using weather data from the Western Regional Climate Center (wrcc.dri.edu)~~and spatially heterogeneous hydraulic conductivity (Figure 6D) generated with the example model input script. The~~.

Water Canyon is unique among the three example sites in that its outflow drains to the ocean. It therefore requires GSFLOW–GRASS to accommodate irregular boundaries (coastlines) by properly assigning boundary conditions and routing flow through them. Users identify ocean pixels by assigning NULL values to them; this causes flow routing from **r.watershed** to stop at the shoreline. To allow flow out of pour-point at the mouth of the river, the immediately downgradient MODFLOW cell can be set as a constant-head boundary, but this cell must be chosen carefully. The finite-difference scheme in MODFLOW dictates that the constant head boundary condition must be supplied along one of the four cardinal directions of the pour-point. Therefore, if the river flows diagonally to the sea, its constant-head boundary must be moved to the closest non-diagonal cell. **v.gsflow.grid** finds the proper constant-head boundary cell to set for the coastal case, as well as for any inland drainage case in which the pour point also requires a downgradient constant-head boundary.

Losing streams such as those in the steep and semi-arid ~~climate can lead to losing streams that may run dry (Jazwa et al., 2016) , which may make GSFLOW simulations susceptible to having entire MODFLOW cells become dry and cause numerical convergence issues. The GSFLOW-GRASS implementation does simulate low streamflow as expected (Figure 6B) , but the domain-builder generated sufficiently thick and hydrologically corrected MODFLOW cells that they maintain water and avoid any computational problems~~Water Canyon catchment often run dry Jazwa et al. (2016) . If this causes MODFLOW cells to lose all of their water, GSFLOW will fail to numerically converge. Thus, the Water Canyon example also serves to demonstrate GSFLOW–GRASS' ability to prevent this problem by (1) incorporating MODFLOW-NWT, which uses a Newton–Raphson solver for increased stability (Niswonger et al., 2011) ; (2) allowing the user to specify an adequately deep MODFLOW discretization in the Settings file (Section 3.1) to supply sufficient water through the dry season; and (3) hydrologically correcting the elevations of coarsened MODFLOW cells to enforce integrated drainage through the stream network. Focusing on the third approach that is specific to the GSFLOW–GRASS toolbox (Section 3.2.2), the narrow and steep Water Canyon requires the same hydrologic corrections that were applied in the Shullcas case, above, to maintain downslope-integrated drainage. Under losing stream conditions, artificially increased channel elevation would steepen the hydraulic head gradient away from the channel and cause it to over-simulate water flow to the surrounding landscape. Therefore, hydrologic correction of the coarse MODFLOW grid is necessary to simulate appropriate head gradients and maintain water in cells, which is further required for any attempt to match stream-gauge records.

The Santa Rosa example demonstrates an application in which GSFLOW–GRASS can be used to investigate and manage erosion associated with hydrological conditions. Erosion of upland areas moves sediment downslope to the areas flanking the stream channel, which contains coarser-grained alluvial sediments. We represented this heterogeneity using a spatially distributed hydraulic conductivity field (Figure 6D) generated with the example model input script included in GSFLOW–GRASS (Section 3.3.3). Figure 6E demonstrates how the post-processing tools can be used to evaluate surface runoff, a ~~concern because of its potential for causing~~ driver of erosion on the island (Schumann et al., 2016). Simulations show precipitation events triggering surface runoff (Figure 6E), which could denude the hillslopes and transport eroded sediment through the drainage network (Figure 6B).

4.3 Cannon River, Minnesota, USA

The Cannon River is a tributary to the ~~Upper~~ upper Mississippi River in Minnesota, USA. Its headwaters cross low-relief uplands that are capped by ~~low hydraulic conductivity~~ 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 uses 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)~~ (Blumentritt et al., 2009; Carson et al., 2018) . This leads to a suite of management concerns related to agricultural nutrients and fine sediments (Czuba and Foufoula-Georgiou, 2014) , and their interactions with both the surface water and the bedrock ~~groundwater~~ aquifer systems that underlie them (Tipping, 2006; Steenberg et al., 2013), thus motivating the need for integrated hydrologic modeling tools.

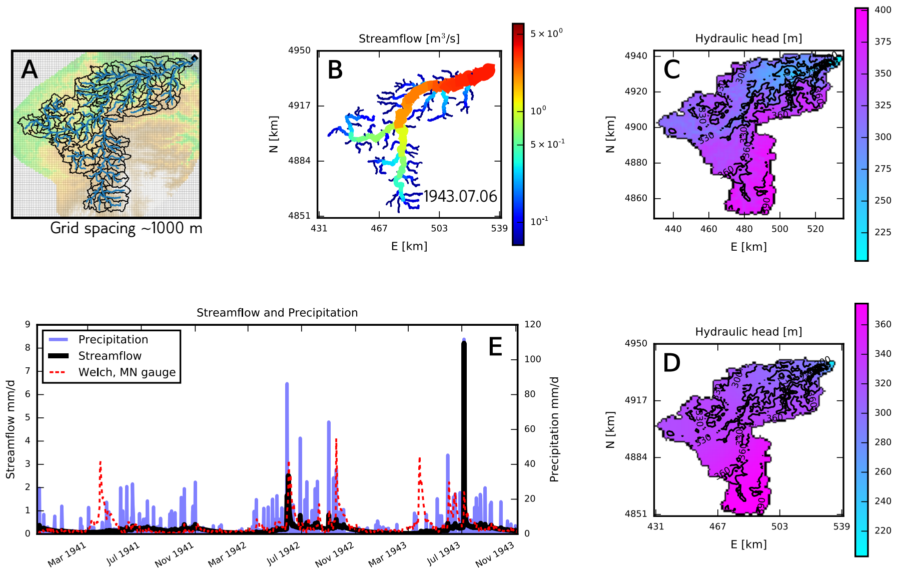

Figure 7. Model based on Cannon River, Minnesota, USA. (**A**) Map with MODFLOW grid, HRU outlines, stream segments (blue), and digital elevation model. (**B**) Simulated discharge after an 11 cm rainfall event. (**C, D**) ~~Two~~ Relatively low-gradient hydraulic head distributions in two MODFLOW layers ~~were implemented to represent~~ representing an upper glacial till unit (low hydraulic conductivity) and ~~the underlying~~ lower fractured carbonate bedrock (higher hydraulic conductivity)~~. In the model~~, ~~this low-relief catchment exhibits~~ with elevation contours (m.a~~shallow water table, except around the river gorge near the outlet~~.s.l.). (**E**) ~~The two-year~~ Three-year hydrograph ~~shows~~ showing uncalibrated discharge simulations matching observations reasonably well during non-peak flood times but ~~failing to capture~~ poorly during many of the actual peaks~~with default model parameters~~.

~~The~~ We implemented GSFLOW–GRASS for the Cannon River watershed using the Minnesota statewide 1 m LiDAR data set (http://www.mngeo.state.mn.us/chouse/elevation/lidar.html), which we resampled to 15 m resolution. We discretized the subsurface of the Cannon River watershed into 1 km MODFLOW grid cells. Meteorological data from nearby Zumbrota, Minnesota was obtained from the Midwestern Regional Climate Center (Adresen et al., 2014) . The flexible GSFLOW–GRASS input builder allows for easy implementation of two MODFLOW layers to represent an upper glacial till unit (low hydraulic conductivity) and the underlying fractured carbonate bedrock (higher hydraulic conductivity).

Covering 3723 km^2, the Cannon River watershed ~~covers~~ is by far the largest of the three model implementations (Table 2) and benefits from the efficiency of the topographically based surface grid and the hydrologic robustness of the grid coarsening method in GSFLOW–GRASS. 17,455,046 flow-routing grid cells, each of which is 225 m^2 in area, were converted to only 610 irregular HRUs of $\geq$10 km^2 in area. For the groundwater domain, the elevation data were coarsened and hydrologically corrected to a 1 km regular MODFLOW grid.

Over the 3723 km^2 ~~with~~ drainage area, there is only 210 m of total relief~~(Table 2), including~~ , and Pleistocene glaciation produced a significant amount of non-integrated drainage that ~~leads to~~ presents very different computational challenges than those in the steep watersheds discussed previously. Much of ~~this watershed's~~the Cannon River watershed's post-glacial topography is characterized by small ~~local~~ localized hills and enclosed basins that have not yet been organized (or integrated) by fluvial erosion into a linked valley network, in which water ~~would flow~~ flows directly to a stream without encountering an enclosed ~~lake or other basin~~depression (such as a lake, wetland, or dry basin). In such settings ~~lacking~~ that lack integrated drainage, ~~simple~~ downslope flow-routing ~~algorithms typically fail, and "pit filling" can~~ and "pit-filling" algorithms that are typically used to build hydrologic model domains (e.g., Bhatt et al., 2014; Maxwell et al., 2017; Gardner et al., 2018) can fail or produce spurious results by inappropriately modifying the real topography. ~~GSFLOW-GRASS routes~~ As described in Section 3.2.1, GSFLOW–GRASS determines surface-water flow using the GRASS ~~GIS's~~ GIS's efficient and accurate **r.watershed** extension, which implements a least-cost path algorithm ~~, which is~~ designed to produce drainage networks that route flow in the long-range path of steepest descent regardless of the degree of local drainage integration. By using **r.watershed** ~~, GSFLOW-GRASS~~ alongside a set of new GRASS-GIS extensions that integrate it into the GSFLOW framework, GSFLOW–GRASS is able to automatically create a topologically correct and linked drainage network in settings that lack integrated drainage for hydrologic model simulations. While this successfully builds the computational domain for the watershed, the user must still put significant effort into adjusting the HRU parameters in the uplands to appropriately partition rainfall, storage, infiltration, and runoff.

~~We implemented GSFLOW-GRASS using the Minnesota state-wide 1 m LiDAR data set (), which we resampled to 15 m resolution. We discretized the subsurface of the Cannon River watershedinto 1 km MODFLOW grid cells. Meteorological data from nearby Zumbrota, Minnesota was obtained from the Midwestern Regional Climate Center (Adresen et al., 2014) .~~ The northern, mid-continental temperate setting makes the Cannon River watershed the example application with the most evenly distributed precipitation across seasons and strongest seasonal temperature differences. In the model, this low-relief catchment generally exhibits low hydraulic head gradients in both MODFLOW layers, except around the river gorge near the outlet, where head levels drop (Figure 7C-D). Comparisons between the simulated streamflow at the watershed outlet and corresponding observations at Welch, MN over the three-year model run reveal that without any parameter calibrations, the model produces realistic discharge during non-peak flood times and during one of the observed peaks during July ~~1942.~~ 1942 (Figure 7E). The severely over-simulated discharge in July 1943 may be evidence for a local convective summer storm system passing over the Zumbrota weather station, which is located outside of the watershed boundary. Recurring failure of the model to capture April discharge indicates that snowmelt-related parameters require adjustment. Once the model is calibrated, which can be ~~greatly facilitated by employing the automated GSFLOW-GRASS~~ facilitated by applying parameter-estimation approaches (e.g., Doherty, 1994; Poeter and Hill, 1998, 1999) together with the automated GSFLOW–GRASS toolkit, results ~~such as those shown in Figure 7~~ can be used to evaluate infiltration from overlying agricultural plots to shallow and low-gradient water tables, as well as subsequent flushing of impacted shallow groundwater into the river channels during major storms, as shown in Figure 7B.

5 Conclusions

To address the need for a fully automated and freely accessible software that handles the complete workflow for implementing complex hydrologic models, we have created ~~GSFLOW-GRASS~~GSFLOW–GRASS, a bundled toolkit for the coupled surface-water and groundwater model GSFLOW, using open-source Python scripts and GRASS GIS commands. ~~GSFLOW-GRASS~~ GSFLOW–GRASS allows 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. In order to create a robust tool that can be widely implemented over diverse hydro(geo)logic settings, we built a set of GRASS GIS extensions that automatically discretizes a topological surface-water flow network that is linked with the underlying gridded groundwater domain. Our fully automated and generalized toolbox advances 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 have demonstrated ~~GSFLOW-GRASS~~ GSFLOW–GRASS using three diverse examples based on topographies and climates from the water-stressed Andes, Santa Rosa Island off the coast of California, USA, and the intensively farmed Upper Midwest region of the United States. The results show that the new and automated GRASS GIS extensions can automatically and consistently build topologically complete linked surface and subsurface flow domains in settings that are typically challenging for standard GIS tools, including steep topographies, irregular coastal boundaries, and low-relief terrains that lack integrated drainage. Although uncalibrated, these examples further demonstrate that ~~GSFLOW-GRASS~~ GSFLOW–GRASS is a flexible tool for investigating the role of 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.

We designed ~~GSFLOW-GRASS~~ GSFLOW–GRASS to strike a balance between direct “out-of-the-box” functionality and full flexibility for customizing model runs. A default implementation can be launched with no programming required by the user to readily produce preliminary uncalibrated simulations that can serve as a springboard for further model-parameter adjustment through the fully commented toolkit scripts. A key feature of ~~GSFLOW-GRASS~~ GSFLOW–GRASS is its use of all open-source software, enabling users anywhere to apply GSFLOW. We believe that the open-source platform will facilitate future toolbox enhancements through efforts by not only the original ~~GSFLOW-GRASS~~ GSFLOW–GRASS developer team, but also new model users. We envision a number of new capabilities to tackle the grand challenge of handling spatial heterogeneity in integrated hydrologic models. Higher resolution land-surface variability could be achieved by further subdividing sub-basins according to vegetation, soil type, or other geographic features to produce HRUs. Obtaining spatially variable information can be facilitated by linking ~~GSFLOW-GRASS~~ GSFLOW–GRASS to existing regional to international databases for meteorology, soil and geologic properties, and land cover. Further calibration of spatially distributed parameters can be carried out by directly setting up ~~GSFLOW-GRASS~~ GSFLOW–GRASS with a flexible inverse modeling code (e.g., Doherty, 1994; Poeter and Hill, 1998, 1999). It is our hope that with its generalized form and open-access, ~~GSFLOW-GRASS~~ GSFLOW–GRASS can become a community tool that continues to grow to better solve hydrologic and water resources problems of both scientific and general management concerns.

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 contributed equally to this work; they 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. Helpful comments from two anonymous referees guided manuscript improvements.

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.

Arge, L., Chase, J. S., Halpin, P., Toma, L., Vitter, J. S., Urban, D., and Wickremesinghe, R.: Efficient flow computation on massive grid terrain datasets, GeoInformatica, 7, 283–313, https://doi.org/10.1023/A:1025526421410, 2003.

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.

Arnold, J. G., Moriasi, D. N., Gassman, P. W., Abbaspour, K. C., and White, M. W.: SWAT: Model use, calibration, and validation, Biological Systems Engineering, 55, 1491 – 1508, 2012.

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.

Baraer, M., Mckenzie, J., Mark, B. G., Gordon, R., Bury, J., Condom, T., Gomez, J., Knox, S., and Fortner, S. K.: Contribution of groundwater to the outflow from ungauged glacierized catchments: A multi-site study in the tropical Cordillera Blanca, Peru, Hydrological Processes, 29, 2561–2581, https://doi.org/10.1002/hyp.10386, 2015.

Barnes, R., Lehman, C., and Mulla, D.: Priority-flood: An optimal depression-filling and watershed-labeling algorithm for digital elevation models, Computers and Geosciences, 62, 117–127, https://doi.org/10.1016/j.cageo.2013.04.024, http://dx.doi.org/10.1016/j.cageo.2013.04.024, 2014.

Beven, K.: A manifesto for the equifinality thesis, Journal of Hydrology, 320, 18–36, https://doi.org/10.1016/j.jhydrol.2005.07.007, 2006.

Bhatt, G., Kumar, M., and Duffy, C. J.: A tightly coupled GIS and distributed hydrologic modeling framework, Environmental Modelling and Software, 62, 70–84, https://doi.org/10.1016/j.envsoft.2014.08.003, http://dx.doi.org/10.1016/j.envsoft.2014.08.003, 2014.

Bhowmik, A. K., Metz, M., and Schäfer, R. B.: An automated, objective and open source tool for stream threshold selection and upstream riparian corridor delineation, Environmental Modelling and Software, 63, 240–250, https://doi.org/10.1016/j.envsoft.2014.10.017, 2015.

Bird, M. I., O'Grady, D., and Ulm, S.: Humans, water, and the colonization of Australia, Proceedings of the National Academy of Sciences, 113, 11 477–11 482, https://doi.org/10.1073/pnas.1608470113, http://www.pnas.org/lookup/doi/10.1073/pnas.1608470113, 2016.

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.

Braun, J. and Willett, S. D.: A very efficient O(n), implicit and parallel method to solve the stream power equation governing fluvial incision and landscape evolution, Geomorphology, 180-181, 170–179, https://doi.org/10.1016/j.geomorph.2012.10.008, http://dx.doi.org/10.1016/j.geomorph.2012.10.008http://linkinghub.elsevier.com/retrieve/pii/S0169555X12004618, 2013.

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.

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 Mid-Continent Drainage Systems, GSA Today, 28, https://doi.org/10.1130/GSATG355A.1, http://www.geosociety.org/gsatoday/science/G355A/abstract.htm, 2018.

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.

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.

Gallagher, M. and Doherty, J.: Parameter estimation and uncertainty analysis for a watershed model, Environmental Modelling & Software, 22, 1000–1020, https://doi.org/10.1016/j.envsoft.2006.06.007, http://linkinghub.elsevier.com/retrieve/pii/S1364815206001629, 2007.

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, 0.

Gardner, M. A., Morton, C. G., Huntington, J. L., Niswonger, R. G., and Henson, W. R.: Input data processing tools for the integrated hydrologic model GSFLOW, Environmental Modelling & Software, https://doi.org/10.1016/j.envsoft.2018.07.020, https://linkinghub.elsevier.com/retrieve/pii/S1364815217311908, 2018.

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.

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.

Heckmann, T., Schwanghart, W., and Phillips, J. D.: Graph theory-Recent developments of its application in geomorphology, Geomorphology, 243, 130–146, https://doi.org/10.1016/j.geomorph.2014.12.024, http://dx.doi.org/10.1016/j.geomorph.2014.12.024, 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.

Jenson, S. K. and Domingue, J. O.: Extracting topographic structure from digital elevation data for geographic information system analysis, Photogrammetric Engineering and Remote Sensing, 54, 1593–1600, https://doi.org/0099-1112/88/5411-1593$02.25/0, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.138.6487&amp;rep=rep1&amp;type=pdf, 1988.

Kinner, D., Mitasova, H., Stallard, R., Harmon, R. S., and Toma, L.: GIS-Based Stream Network Analysis for the Upper Río Chagres Basin, Panama, in: The Río Chagres, Panama: A Multidisciplinary Profile of a Tropical Watershed, edited by Harmon, R. S., pp. 83–95, Springer-Verlag, Berlin/Heidelberg, https://doi.org/10.1007/1-4020-3297-8_6, http://link.springer.com/10.1007/1-4020-3297-8_6, 2005.

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.

Leonard, L. and Duffy, C. J.: Essential Terrestrial Variable data workflows for distributed water resources modeling, Environmental Modelling & Software, 50, 85–96, https://doi.org/10.1016/j.envsoft.2013.09.003, http://linkinghub.elsevier.com/retrieve/pii/S1364815213001941, 2013.

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.

LeVeque, R. J.: Finite volume methods for hyperbolic problems, vol. 31, Cambridge university press, 2002.

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.

Luzio, M. D., Arnold, J. G., and Srinivasan, R.: A GIS Coupled hydrological model system for the watershed assessment of agricultural nonpoint and point sources of polution, Transactions in GIS, 8, 113–136, https://doi.org/10.1111/j.1467-9671.2004.00170.x, 2006.

Magalhães, S. V. G., Andrade, M. V. A., Franklin, W. R., and Pena, G. C.: A linear time algorithm to compute the drainage network on grid terrains, Journal of Hydroinformatics, 16, 1227, https://doi.org/10.2166/hydro.2013.068, http://jh.iwaponline.com/cgi/doi/10.2166/hydro.2013.068, 2014.

Maidment, D. R. and Morehouse, S.: Arc Hydro: GIS for water resources, vol. 1, ESRI, Inc., 2002.

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.

Maxwell, R. M., Kollet, S. J., Smith, S. G., Woodward, C. S., Falgout, R. D., Ferguson, I. M., Engdahl, N., Condon, L. E., Hector, B., Lopez, S., Bearup, L., Jefferson, J., Collins, C., Graaf, I. D., Pribulick, C., Baldwin, C., Bosl, W. J., Hornung, R., and Ashby, S.: ParFlow User's Manual, Integrated Ground- Water Modeling Center Report GWMI 2016-01, 2017.

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.

Miliaresis, G. and Delikaraoglou, D.: Effects of percent tree canopy density and DEM misregistration on SRTM/NED vegetation height estimates, Remote Sensing, 1, 36–49, https://doi.org/10.3390/rs1020036, 2009.

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.

QGIS Development Team: QGIS Geographic Information System, http://qgis.osgeo.org, 2013.

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.

Razavi, S. and Gupta, H. V.: What do we mean by sensitivity analysis? The need for comprehensive characterization of "global" sensitivity in Earth and Environmental systems models, Water Resources Research, 51, 3070–3092, https://doi.org/10.1002/2014WR016527.Received, 2015.

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.

Rossi, M. and Reichenbach, P.: LAND-SE: A software for statistically based landslide susceptibility zonation, version 1.0, Geoscientific Model Development, 9, 3533–3543, https://doi.org/10.5194/gmd-9-3533-2016, 2016.

Sangireddy, H., Stark, C. P., Kladzyk, A., and Passalacqua, P.: GeoNet: An open source software for the automatic and objective extraction of channel heads, channel network, and channel morphology from high resolution topography data, Environmental Modelling & Software, 83, 58–73, https://doi.org/10.1016/j.envsoft.2016.04.026, http://linkinghub.elsevier.com/retrieve/pii/S1364815216301219, 2016.

Schumann, R. R., Pigati, J. S., and McGeehin, J. P.: Fluvial system response to late Pleistocene-Holocene sea-level change on Santa Rosa Island, Channel Islands National Park, California, Geomorphology, 268, 322–340, https://doi.org/10.1016/j.geomorph.2016.05.033, http://dx.doi.org/10.1016/j.geomorph.2016.05.033, 2016.

Schwanghart, W. and Scherler, D.: Short Communication: TopoToolbox 2 – MATLAB-based software for topographic analysis and modeling in Earth surface sciences, Earth Surface Dynamics, 2, 1–7, https://doi.org/10.5194/esurf-2-1-2014, http://www.earth-surf-dynam.net/2/1/2014/, 2014.

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.

Somers, L. D., McKenzie, J. M., Zipper, S. C., Mark, B. G., Lagos, P., and Baraer, M.: Does hillslope trenching enhance groundwater recharge and baseflow in the Peruvian Andes?, Hydrological Processes, 32, 318–331, https://doi.org/10.1002/hyp.11423, 2018.

Song, X., Zhang, J., Zhan, C., Xuan, Y., Ye, M., and Xu, C.: Global sensitivity analysis in hydrological modeling: Review of concepts, methods, theoretical framework, and applications, Journal of Hydrology, 523, 739–757, https://doi.org/10.1016/j.jhydrol.2015.02.013, http://dx.doi.org/10.1016/j.jhydrol.2015.02.013, 2015.

Srinivasan, R. and Arnold, J. G.: Integration of a basin-scale water quality model with GIS, Journal of the American Water Resources Association, 30, 453–462, https://doi.org/10.1111/j.1752-1688.1994.tb03304.x, http://doi.wiley.com/10.1111/j.1752-1688.1994.tb03304.x, 1994.

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.

Tejedor, A., Longjas, A., Zaliapin, I., and Foufoula-Georgiou, E.: Delta channel networks: 1. A graph-theoretic approach for studying connectivity and steady state transport on deltaic surfaces, Water Resources Research, 51, 3998–4018, https://doi.org/10.1002/2014WR016577, http://doi.wiley.com/10.1002/2014WR016577, 2015.

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.

Tian, Y., Zheng, Y., and Zheng, C.: Development of a visualization tool for integrated surface water-groundwater modeling, Computers and Geosciences, 86, 1–14, https://doi.org/10.1016/j.cageo.2015.09.019, http://dx.doi.org/10.1016/j.cageo.2015.09.019, 2016.

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.

Wang, D., Liu, Y., and Kumar, M.: Using nested discretization for a detailed yet computationally efficient simulation of local hydrology in a distributed hydrologic model, Scientific Reports, 8, 1–13, https://doi.org/10.1038/s41598-018-24122-7, http://dx.doi.org/10.1038/s41598-018-24122-7, 2018.

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.

Wickert, A. D., Martin, J. M., Tal, M., Kim, W., Sheets, B., Paola, C., Paola, S. C., Sheets, B., Paola, C., Paola, S. C., Sheets, B., and Paola, C.: River channel lateral mobility: Metrics, time scales, and controls, Journal of Geophysical Research: Earth Surface, 118, 396–412, https://doi.org/10.1029/2012JF002386, http://doi.wiley.com/10.1029/2012JF002386, 2013.

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.

úri, 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.