RESPONSE TO REVIEWER 1

<u>Boldface:</u> **Reviewer 1's original comments and additional clarifications (latter added to the corresponding original comment)**
<u>Italics:</u> *Our response*

We thank the referee for their time in reviewing our manuscript and providing feedback.

General comments: The manuscript presents the development of a suite of tools for preparing the input, submitting the simulation runs, and visualizing the output of the groundwater–surface-water coupled GSFLOW model. The proposed suite of tools is developed exploiting the functionalities of the open-source GIS software GRASS and ad-hoc Python scripting. Authors tested the developed toolkit presenting test cases based on three catchments having different physiographic features. The manuscript is generally well written and with a logical and easy-to-follow structure. While I concur with the authors on the potential of such kind of efforts to encourage the use of complex surface-subsurface coupled hydrological models, I question the actual novelty and technical advancements presented in their work. Besides a suite of GIS extensions and scripts, the manuscript does not propose new technical solutions for the problem at-hand. For this reason, and for those elaborated below, I consider this contribution not suitable for GMD standard.

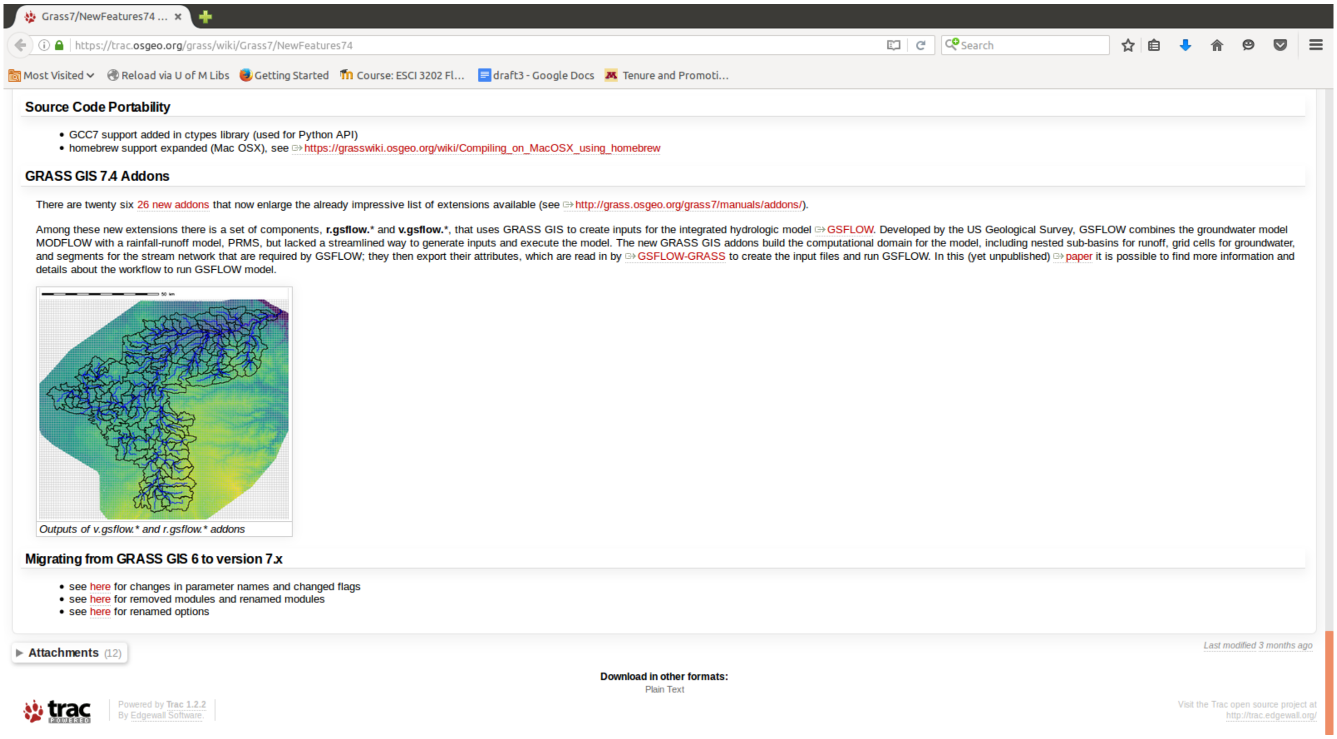
(Additional clarification:) I evaluated the paper not suitable for GMD for the lack of novelty and technical advancements. I did not question the utility itself of the proposed toolkit and I did not express any issue concerning the fit of the subject addressed in this work with the scope of the journal.

We are glad the reviewer found that our manuscript was generally well-written and fits within the GMD scope, but we are obviously very disappointed that the referee did not consider it to be suitable for GMD standard. Our work does offer new technical solutions for making integrated hydrologic modeling more accessible; this review brought to our attention that the first manuscript version indeed failed to explain these novel aspects and technical contributions and instead focused too much on simply documenting the contents of the toolbox. We appreciate that the reviewer raised this issue, and we have substantially revised the manuscript to address this serious shortcoming in the original presentation.

In particular, we first clarified that while some of the individual scripting components within the toolbox may appear straightforward, our work's innovation is the entire bundled package. Our substantially edited Introduction now emphasizes that existing software for integrated hydrologic models fail to provide freely accessible toolkits that fully cover pre- to post-processing steps (p. 2 Lines 22-30), and that GSFLOW-GRASS addresses that critical gap stymieing the use of integrated hydrologic models (p.2 Line 31- p. 3 Line 6).

We then explained that the major technical advancement of our work was to create a new set of GRASS-GIS tools that can robustly and automatically generate surface and subsurface model domains suitable for hydrological modeling – these are critical for GSFLOW-GRASS to be widely applicable to a diverse range of hydro(geo)logical settings. We now realize that the original manuscript version documented these new GRASS GIS extensions but provided almost no background on the challenges of creating robust and automated tools, which have led to a general unavailability of such solutions predating our toolbox. A new paragraph has been added to the Introduction to present the technical advancements with these GRASS GIS extensions (p. 3 Lines 7-21). Further, we have entirely re-written Section 3.2 on the GRASS GIS domain builder (p. 10-12), so that it now explicitly describes what was implemented to solve specific known problems with stream network delineation. Finally, we also made major changes to Section 4 on the Examples, in order to explain how each example demonstrates a different strength and capability of the domain builder (specifically, p. 21 Lines 3-10 for Shullcas, p. 21 Line 32- p. 22 Line 4 for Santa Rosa, and p. 23 Line 10- p. 24 Line 4 for Cannon River). These examples demonstrate how GSFLOW-GRASS handles known challenges with various degrees of drainage integration, landscape relief, and grid resolution, as well as the presence of irregular coastal boundaries.

The technical advancements of our GRASS GIS tools were recently highlighted as a new release feature on the GRASS GIS website: https://trac.osgeo.org/grass/wiki/Grass7/NewFeatures74 (including a figure with our Cannon River watershed example) – see the below screenshot.



Source Code Portability

- GCC7 support added in ctypes library (used for Python API)
- homebrew support expanded (Mac OSX), see https://grasswiki.osgeo.org/wiki/Compiling_on_MacOSX_using_homebrew

GRASS GIS 7.4 Addons

There are twenty six 26 new addons that now enlarge the already impressive list of extensions available (see http://grass.osgeo.org/grass7/manuals/addons/).

Among these new extensions there is a set of components, **r.gsflow.*** and **v.gsflow.***, that uses GRASS GIS to create inputs for the integrated hydrologic model GSFLOW. Developed by the US Geological Survey, GSFLOW combines the groundwater model MODFLOW with a rainfall-runoff model, PRMS, but lacked a streamlined way to generate inputs and execute the model. The new GRASS GIS addons build the computational domain for the model, including nested sub-basins for runoff, grid cells for groundwater, and segments for the stream network that are required by GSFLOW; they then export their attributes, which are read in by GSFLOW-GRASS to create the input files and run GSFLOW. In this (yet unpublished) paper it is possible to find more information and details about the workflow to run GSFLOW model.

Outputs of v.gsflow. and r.gsflow.* addons*

Migrating from GRASS GIS 6 to version 7.x

- see here for changes in parameter names and changed flags
- see here for removed modules and renamed modules
- see here for renamed options

Attachments (12)

Last modified 3 months ago

Download in other formats:
Plain Text

Powered by **Trac 1.2.2**
By Edgewall Software.

Visit the Trac open source project at
http://trac.edgewall.org/

In addition to major revisions to the Introduction, GRASS GIS Domain Builder section, and Examples section, we essentially re-wrote the Abstract and Conclusion to highlight these new and technical contributions.

Specific comments:

1. In presenting/justifying their work, I think authors overlooked a bit too much the key technical issues preventing the widespread use of complex, physically based surface-subsurface coupled hydrological models in a decision-making framework. Here, I would argue that preparing the input is certainly a necessary and important step in the modeling exercise but not the most challenging one. In fact, if we agree that computationally efficient and numerically stable codes are needed to “promote science-driven decision making” then ad-hoc tools allowing a dynamic (e.g. in-situ visualization) inspection of such physical and numerical model response are probably much more needed, especially when we approach big-data problems. Saying that, I do not see the positioning of the effort presented in this manuscript with respect to these grand challenging tasks.

(Additional clarification:) I highlighted some of the grand challenges (e.g., big-data problems) that, in my opinion, modelers are facing when performing large-scale high-resolution surface-subsurface coupled simulations. In this context, in-situ visualization (i.e., the use of libraries to dynamically connect running simulations and graphical outputs) is of particular interest in the geoscience community. My concern was that the paper did not even mention/discuss how the methodology they are proposing reconcile with such grand challenge.

We believe that the need to create long model input files does in fact present a critical challenge for many potential users who may lack the necessary software skills or who might wish to carry out initial model tests before committing time to its use. In support of the value of our toolbox, we would like to share that over just April 9 to 22 (the maximum length of time tracked by GitHub), our GSFLOW-GRASS repository received 173 *views and 22 unique visitors, and one user from a major research university sent us an email that opened with "Thank you so much for sharing the GSFLOW-GRASS toolkit. This toolkit really relieves my struggle of preparing inputs." - and all of this is with absolutely no effort to advertise our toolkit.*

However, we do acknowledge that there are other grand challenges to integrated hydrologic modeling, and we appreciate the reviewer's suggestion about in-situ visualization. In response, we have expanded our toolbox to include an additional tool "plotGSFLOWTimeSeries_Runtime.py," which is now incorporated into our Run script to generate runtime time series plots of simulated discharge and precipitation. This new capability is described in the revised manuscript on p. 17 Lines 12-13 and p. 18 Lines 10-13.

2. The outcome of the presented developments is clearly reflected in the results section. Here, authors describe three test cases illustrating the physical settings of each study area and discussing the potential outcome of a surface-subsurface coupled modeling approach. However, these results appear the repetition of the same exercise without much insight on the novelty of the proposed approach. For instance one could argue that such kind of plots can be simply obtained with some visualization scripts developed from scratch.

(Additional clarification:) I questioned the insights gained from the three test cases. Authors reply that each of them demonstrates particular technical challenges solved by the proposed toolkit where 'other' approaches would fail. If this is the case, you need to provide evidence, from a simple visual inspection of Figure 5-6-7 I do not see it.

A user can indeed develop from scratch similar visualization scripts, but we believe that the need to do so presents a major impediment to many potential users who may lack the programming knowledge or may not be able or wish to invest the time for it. Our toolbox includes pre- and post-processing capabilities that make the GSFLOW model widely accessible.

We do acknowledge, however, that the original manuscript version presented the 3 examples in a way that did not describe how each one presents a particular challenge that the new GRASS GIS extensions address. As mentioned earlier in this response, we addded paragraphs to each example to do so, specifically: p. 21 Lines 3-10 for Shullcas, p. 21 Line 32- p. 22 Line 4 for Santa Rosa, and p. 23 Line 10- p. 24 Line 4 for Cannon River. These aspects are also summarized in the Conclusions: "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." (p. 24 Lines 31-33).

We further realized that we should have more clearly highlighted the types of hydrologic / hydrogeologic processes of management concern that can be evaluated with aid of GSFLOW-GRASS through each example; we have edited the last paragraph of each example to better express these types of processes and how they are depicted with the GSFLOW-GRASS visualization tools. These processes are also summarized in the edited Conclusion: "these examples further demonstrate that 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." (p. 25 Lines 1-3).

3. In a similar vein to the previous point, at the end of the introduction authors argue that the developments of such automated toolkit will enable rigorous testing. Absolutely true but a concrete path forward and tangible results are not presented in this context. Wouldn't it be an interesting way to demonstrate the utility of such kind of tools?

As we discussed in our response to the previous comment 2., we realize that our original manuscript version failed to adequately explain how the 3 examples demonstrated the utility of GSFLOW-GRASS. We have *substantially edited our manuscript to now explain how each of these examples showcase a particular capability of the domain builder as well as a different scientific and/or resource management concern affected by groundwater-surface interactions that can be probed with the aid of GSFLOW-GRASS (see manuscript lines referenced above). As a concrete path forward, we also suggest future tests of the performance of ungridded surface domains with GSFLOW-GRASS (p. 9 Lines 6-7), and we list potential future extensions of GSFLOW-GRASS in the Conclusion (p. 25 Lines 9-17).*

4. In several parts of the manuscript, authors refer to a similar work, i.e., Gardner et al., which is currently under review for another journal. As the content of the cited work cannot be evaluated, these statements are unverifiable by the reader/reviewer, which is obviously not acceptable. Moreover, considering the potential overlap between the two contributions, as also acknowledged by the authors, it is not possible to weight the actual contribution of this work. For instance, one may ask if moving from ArcGis to GRASS or using ungridded versus gridded data would be enough to motivate an additional publication.

(Additional clarification:) I raised the issue of a cited publication, which is currently under review for another journal. Authors' argumentation is that the work received positive comments and it will be likely out very soon. At this time it is not. Therefore, it is not possible for any reviewer or person eager to comment on the manuscript to have an idea on the content of the cited work. In other words, being aware of these positive comments on the contribution, you should have included in the discussion later in the review process.

We now recognize that it is unreasonable to expect a reader to follow detailed comparisons with an unpublished and unavailable manuscript. One of our co-authors, Rich Niswonger, is also a co-author of the Gardner et al. submitted manuscript (as well as one of the GSFLOW developers at USGS). He reports that the manuscript is still in re-review at this time. Because the Gardner et al. work is not actually central to GSFLOW-GRASS, we minimized our discussion of that work – mostly in the Introduction and the "User-specified settings and model inputs" section. We now only mention it as one of the other software options to facilitate integrated hydrologic model implementation that do not offer a complete pre- to post-processing set of tools. We view it as a real benefit for the community to have these two new packages with different features (differences in discretization, handling of input data, availability of post-processing tools, and software platforms), so that users can choose the one most suitable for their application. Rich Niswonger's role as co-author has not been to develop the GSFLOW-GRASS software, but it has been to ensure that GSFLOW-GRASS is not overly duplicative of the package by Gardner et al. (of which he is a developer), and that GSFLOW-GRASS is constructed in a way that the USGS considers will be effective for increasing the accessibility of GSFLOW. The multiple softwares (free and proprietary) available for implementing MODFLOW serve as an example that having more than one software package for a model can be valuable for supporting an extensive user-base.

5. It appears that for some of the most critical parameters (e.g., Manning's parameter) authors present their approach referring to homogeneous values. In so doing, they advocate that field data on channel geometries come in a variety of forms difficult to accommodate in a generalized approach. Wouldn't it be the motivating reason for such geoscientific developments as the one presented here? Data fusion tools are in my opinion the key for facilitating the coherent ingestion of large source of information into a distributed model input data structure. An example along this line is represented by the work of Leonard and Duffy, 2013.
References: Lorne Leonard, Christopher J. Duffy, Essential Terrestrial Variable data workflows for distributed water resources modeling, Environmental Modelling & Software,50, 85-96, 10.1016/j.envsoft.2013.09.003, 2013.

Our original GSFLOW-GRASS version did include an option for spatially heterogeneous hydraulic conductivity inputs, which was implemented in the Santa Rosa Island example. However, we agree with the reviewer that a heterogeneous channel width and Manning's n parameter would also be important to include, and in response, we modified the toolbox to accommodate this through the Settings file (as described on p. 9 Line 29 and on p. 17

Line 5).

We acknowledge the value of linking integrated modeling with existing databases for model inputs, but we consider this beyond the scope of our current work, which aims to provide a generalized solution for implementing GSFLOW-GRASS. We reference software tools that do fuse data products with hydrologic models, including Leonard and Duffy (2013) as suggested by the reviewer (p. 10 Lines 3-5); we then point out that these databases are generally only available in observation-rich places and thus we do not include any in the first GSFLOW-GRASS version, which serves as a general basis for further development (p. 10 Lines 7-8). Our revised conclusion discusses future extensions of GSFLOW-GRASS to include links to spatial databases to generate model inputs (p. 25 Lines 12-13).

Although GSFLOW-GRASS currently does not offer spatially heterogeneous solutions for inputs beyond hydraulic conductivity and Manning's n, we created a new GRASS GIS tool, v.gsflow.mapdata, in response to the reviewer's valid concern about it. This tool 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 helps users add data from any source to the GSFLOW-GRASS data structures for input into the model. The new v.gsflow.mapdata tool is presented on p. 9 Line 30- p. 10 Line 8. Throughout the rest of the revised manuscript, we also mention how this tool can be implemented to create specific spatially distributed inputs, including the climate inputs and soil / land-cover parameters (p. 12 Line 26, p. 14 Lines 19 and 29, p. 15 Lines 16 and 29, and p. 17 Line 1).

Technical corrections:

1. Authors argue that models using triangulated irregular networks show better water balance performance over steep catchments. This is a quite interesting statement but ad-hoc citation is needed to substantiate this.

We realize that we left out some details and should have specified that TINs show better water balance performance IF they are implemented with the finite volume method (because the finite volume method is mass-conserving), and that TINs cover complex surface domain more efficiently (fewer units) than grid cells. We edited the text to say all of this on p. 4 Line 15-16.

2. According to the author's opinion, PRMS does not implement Richards equation but instead applies an 'efficient' calculation to determine input and output for HRU. What's the meaning of 'efficient' here?

By "efficient," we mean computationally fast. We clarified this on p. 6 Lines 5-6.

3. I do not see the precipitation lines in Figure 5-6-7.

As we mentioned in our preliminary response to the reviewer: we see the blue precipitation lines clearly in these figures. We are unsure why they do not appear for the reviewer and wonder if there is an issue with the file conversion. If more information could be provided (e.g., do the blue lines fail to appear at all, or do they appear but just not clearly?), we can address it.

RESPONSE TO REVIEWER 2

<u>Boldface:</u> **Reviewer 2's original comments**
<u>Italics:</u> Our response

We thank the referee for their time in reviewing our manuscript and providing feedback.

<u>General comments:</u>
Ng et al. present a GIS-based tool, GSFLOW-GRASS, that prepares input and runs the USGS hydrologic model GSFLOW. The authors provide comprehensive description of the GIS-based software, as well as the USGS hydrologic model. The paper is well-organized and well-written. As a modeler, I highly appreciate the authors' efforts in developing such tools, because "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". However, I do feel that GSFLOW-GRASS has limited capability in handling spatially-distributed, realistic input data (see specific comments #2). All examples are shown without any measured discharge data. I acknowledge that model calibration is beyond the scope of this study, but it would be helpful if measured data could be shown, to demonstrate that the generated input can yield reasonable, if not accurate predictions. The authors also need to do a better job describing what are the "substantial new/novel concepts, ideas, or methods" in developing GSFLOW-GRASS, as required by GMD.

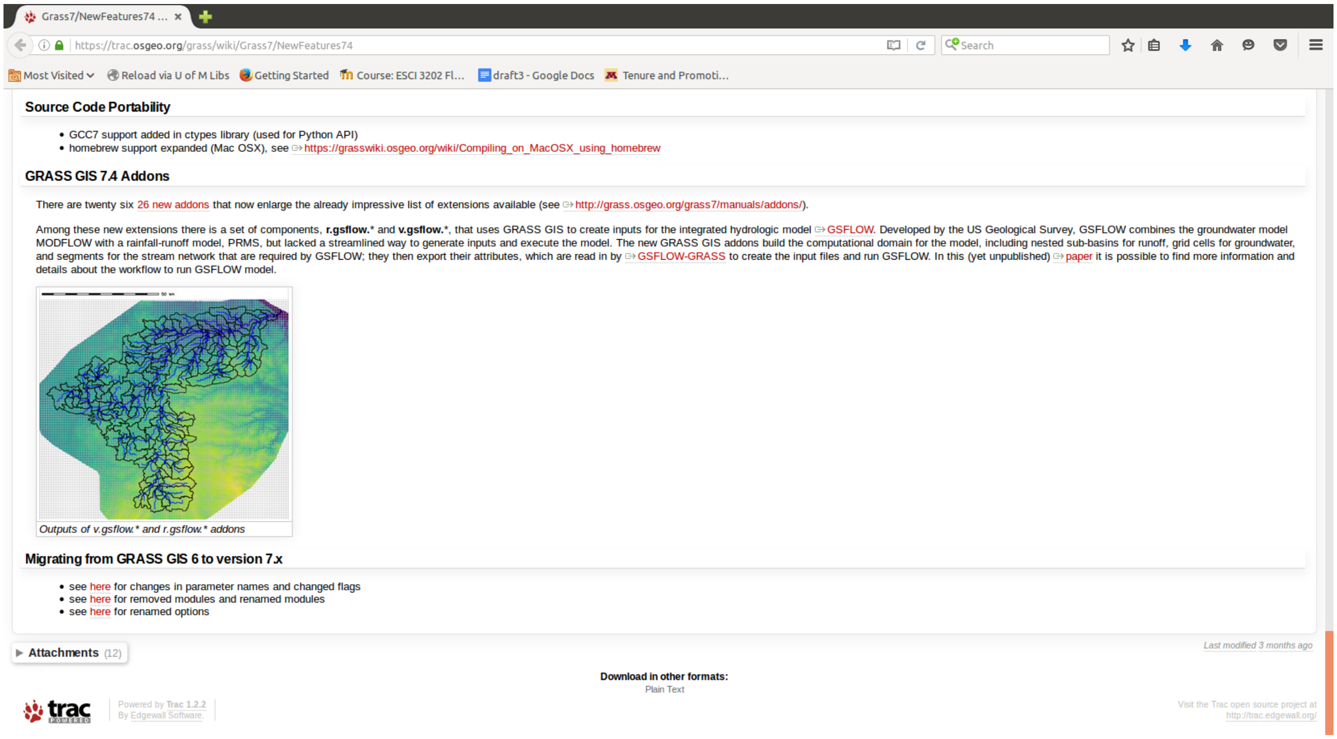
We are glad that the reviewer found our manuscript to be generally well-organized and appreciated our effort to make integrated hydrologic modeling more accessible. Indeed, we believe model-users are already seeing the utility of our work; just over April 9 to 22 (the maximum length of time tracked by GitHub), our GSFLOW-GRASS repository received 173 views and 22 unique visitors, and one user from a research university sent us an email that opened with "Thank you so much for sharing the GSFLOW-GRASS toolkit. This toolkit really relieves my struggle of preparing inputs." - and all of this is with absolutely no effort to advertise our toolkit.

After this review, we do now recognize that our original manuscript version failed to adequately explain the novel aspects and new technical advances provided by our toolbox. We appreciate that the reviewer raised this issue, and we have substantially revised the manuscript to address this serious shortcoming in the presentation.

In particular, we first clarified that while some of the individual scripting components within the toolbox may appear straightforward, our work's innovation is the entire bundled package. Our substantially edited Introduction now emphasizes that existing software for integrated hydrologic models fail to provide freely accessible tool-sets that fully cover pre- to post-processing steps (p. 2, lines 22-30), and that GSFLOW-GRASS addresses that critical gap stymieing the use of integrated hydrologic models (p.2 lines 31-34, p. 3 lines 1-6).

We then explained that the major new method advancement was to create a new set of GRASS-GIS tools that can robustly and automatically generate surface and subsurface model domains suitable for hydrological modeling, which was critical for GSFLOW-GRASS to be widely applicable to a diverse range of hydro(geo)logical settings. We now realize that the original manuscript version documented these new GRASS GIS extensions but provided almost no background on the challenges of creating robust and automated tools, which has led to a general unavailability of such solutions predating our toolbox. A new paragraph has been added to the Introduction to present this technical advancement (p. 3 Lines 7-21). Further, we have entirely re-written Section 3.2 on the GRASS GIS domain builder (p. 10-12), so that it now explicitly describes what was implemented to solve specific known challenges for stream network delineation. Finally, we also made major changes to Section 4 on the Examples, in order to explain how each example demonstrates a different strength and capability of the domain builder (specifically, p. 21 Lines 3-10 for Shullcas, p. 21 Line 32- p. 22 Line 4 for Santa Rosa, and p. 23 Line 10- p. 24 Line 4 for Cannon River). These examples demonstrate how GSFLOW-GRASS handles known challenges with various degrees of drainage integration, landscape relief, and grid resolution, as well as the presence of irregular coastal boundaries.

The new technical advancements of our GRASS GIS tools were recently highlighted as a new release feature on the GRASS GIS website: https://trac.osgeo.org/grass/wiki/Grass7/NewFeatures74 (including a figure with our Cannon River watershed example) – see screenshot below.



Source Code Portability

- GCC7 support added in ctypes library (used for Python API)
- homebrew support expanded (Mac OSX), see https://grasswiki.osgeo.org/wiki/Compiling_on_MacOSX_using_homebrew

GRASS GIS 7.4 Addons

There are twenty six 26 new addons that now enlarge the already impressive list of extensions available (see http://grass.osgeo.org/grass7/manuals/addons/).

Among these new extensions there is a set of components, **r.gsflow.*** and **v.gsflow.***, that uses GRASS GIS to create inputs for the integrated hydrologic model GSFLOW. Developed by the US Geological Survey, GSFLOW combines the groundwater model MODFLOW with a rainfall-runoff model, PRMS, but lacked a streamlined way to generate inputs and execute the model. The new GRASS GIS addons build the computational domain for the model, including nested sub-basins for runoff, grid cells for groundwater, and segments for the stream network that are required by GSFLOW; they then export their attributes, which are read in by GSFLOW-GRASS to create the input files and run GSFLOW. In this (yet unpublished) paper it is possible to find more information and details about the workflow to run GSFLOW model.

Outputs of v.gsflow. and r.gsflow.* addons*

Migrating from GRASS GIS 6 to version 7.x

- see here for changes in parameter names and changed flags
- see here for removed modules and renamed modules
- see here for renamed options

Attachments (12)

Last modified 3 months ago

Download in other formats:
Plain Text

Powered by Trac 1.2.2
By Edgewall Software.

Visit the Trac open source project at
http://trac.edgewall.org/

In addition to major revisions to the Introduction, GRASS GIS Domain Builder section, and Examples section, we essentially re-wrote the Abstract and Conclusion to highlight these new and technical contributions.

The reviewer also brought up an excellent point about the importance of spatially distributed, realistic input data. This prompted us to add new capabilities to the toolbox to accommodate spatially variable inputs, which we describe in greater detail under "Specific comment #2."

We appreciate the reviewer acknowledging that model calibration is beyond the scope of this work. However, the reviewer's comment about comparing against observations made us realize that we needed to be clearer that the aim of the examples is not simulate realistic results for each site but is instead to demonstrate the robustness of the domain builder for a range of settings (as pointed out above) and the types of processes that can be explored with GSFLOW-GRASS. We re-wrote the opening of the Examples section to reflect this (p. 18 Line 32- p. 19 Line 5), and we added a brief discussion of how the automated GSFLOW-GRASS toolbox jump-starts the model implementation and thus facilitates additional calibration and sensitivity analysis (p. 19 Line 18- p. 20 Line 7).

Although the examples are not meant to serve as realistic results, we do recognize the value of comparing our example simulations against actual observations. The Cannon River site is the only one of the 3 example sites that has publicly available discharge data, and so we now show it in our revised Figure 7. In addition to discussing where the uncalibrated model does reasonably match observations, we also point out how the major discrepancies found in the default implementation can be useful for guiding parameter calibration (p. 24 Lines 9-15).

Specific comments:

1. Besides GSFLOW-GRASS and Gardner et al. (2017), there is another for-free software by Earthfx that can generate the inputs for GSFLOW. It is surprising that the authors did not review or describe this

software. What are the differences between GSFLOW-GRASS and Earthfx software?

We were aware of the Earthfx software and did mention it in the original manuscript ("However, with the exception of a single for-fee software product by Earthfx (http://www.earthfx.com/), these tools neither help users conceptually link the different process domains represented in PRMS and MODFLOW nor generate the files that link them." original p. 2 Lines 21-23). The reviewer wrote that Earthfx is "for-free" but we wonder if that is a typo and is supposed to be "for-fee"? Looking again at the Earthfx website, we do not see anywhere that one can freely download the software (http://www.earthfx.com/VIEWLOG/Components/VLGSFLOW.aspx) and only see a page pointing interested users to call for pricing information (http://www.earthfx.com/VIEWLOG/Details/Pricing.aspx says: "Call for pricing," "University discount 30%" for the VIEWLOG software, which includes VL-GSFLOW).

Given that the software is fee-based and cannot be accessed without contacting the vendor, we found it difficult to review in detail. We did add to the text that Earthfx offers "full support" for GSFLOW, specifically through its "VIEWLOG" software package (p. 2 Lines 25-26). We then point out that "the community still lacks a free and complete package spanning pre- to post-processing for heterogeneous surface and subsurface domains" (p. 2 Lines 27-28), motivating the development of GSFLOW-GRASS.

2. The GSFLOW-GRASS tool has limited capability in handling spatially-distributed, realistic input data. For example, P10, L24 "In its current form, v.gsflow.segments . . . allows the user to set a single channel width and Manning's n (in-channel roughness coefficient for flow resistance) across the whole domain;" P13, L29 "we do provide a script for uniformly applying a single climate data series over all HRUs to create climate_hru files;" and P14, L24 "most parameter values in printPRMSparamfile.py are preset. . .. This includes various soil and land-cover inputs, such as soil_type, cov_type, transp_end, and pt_alpha." There are GIS-based hydrologic model input tools that can take all different types of GIS input to generate spatially-distributed input data from national data-base or in situ measurements. For example, PIHMgis (Bhatt, G., Kumar, M. and Duffy, C. J., 2014: A tightly coupled GIS and distributed hydrologic modeling framework, Environmental Modelling & Software. 62, 70—84.). The spatially-uniform approach and the preset default parameter values may prohibit GSFLOW from generating accurate predictions.

Our original GSFLOW-GRASS version did include an option for spatially heterogeneous hydraulic conductivity, which was implemented in the Santa Rosa Island example. However, we agree with the reviewer that the general lack of ability for our original toolbox to generate spatially variable data inputs (beyond hydraulic conductivity) was a weakness in the original submission. To address this, we modified the toolbox to accommodate heterogeneous channel width and Manning's n parameter through the Settings file (as explained on p. 9 Line 29 and on p. 17 Line 5). For other inputs, we created a new GRASS GIS tool, v.gsflow.mapdata, which 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 incorporate data from any source to the GSFLOW-GRASS data structures for input into the model. The new v.gsflow.mapdata tool is presented on p. 9 Line 30- p. 10 Line 8. Throughout the rest of the paper, we also mention how this tool can be implemented to create specific spatially distributed inputs, including the climate inputs and soil / land-cover parameters mentioned by the reviewer (p. 12 Line 26, p. 14 Lines 19 and 29, p. 15 Lines 16 and 29, and p. 17 Line 1).

For linking integrated hydrologic modeling with existing databases for model inputs, we do see the clear utility but consider this to be beyond the scope of our current work, which aims to provide a generalized solution for implementing GSFLOW-GRASS. We reference software tools that focus on this capability, including Bhatt et al. (2014) as suggested by the reviewer (p. 10 Lines 3-5); we then point out that these databases are typically only available in observation-rich places and thus we do not include it in the first GSFLOW-GRASS version, which provides a general basis for further development (p. 10 Lines 7-8). Our revised conclusion discusses future extensions of GSFLOW-GRASS to include links to spatial databases to generate model inputs (p. 25 Lines 11-

12).

3. Page 3, L10 The authors state that triangulated irregular networks have better water balance performance. Why is that?

We realized that we left out some details and should have specified that triangulated irregular networks (TINs) show better water balance performance IF they are implemented with the finite volume method (because the finite volume method is mass-conserving), and that TINs cover complex surface domain more efficiently (fewer units) than grid cells. We edited the text to say all of this on p. 4 Line 15-16.

4. I am interested in the spin-up process described between L6 and L11 on Page 9. The authors describe the initial conditions as preliminary steady-state initial conditions. Usually the spin-up process is aimed to bring models to steady-state. If so, what is the difference between the before- and after-spin-up initial conditions?

We realize we could have explained the steady-state vs. spin-up process much more clearly. The preliminary steady-state run is just for the MODFLOW (groundwater) component; the steady-state groundwater head results are then used to initialize the spin-up for the fully coupled (surface AND subsurface model domains). We edited the text to better explain this on p. 10 Lines 13-17.

5. Some of the parameters are shown without their definitions, for example pref_flow_den and sat_threshold. It would be add definitions.

We added Table 1 to define all GSFLOW parameters mentioned in the manuscript.

Technical comments:

1. Figure 2 caption "Duncan runoff and fast interflow occurs in the preferential-flow reservoir." Should be "occur."
We implemented this edit.

2. P10 L4 "This approach is complementary to the grid-cell HRU approach of (Gardner et al., 2017)." \citet command should be used instead of \citep.
The text was edited in the revision and no longer has this issue.

3. P22 L8 "This allow users . . .." Should be "allows."
We implemented this edit.

RESPONSE TO REVIEWER 3

Boldface: Reviewer 3's original comments
Italics: Our response

We thank the referee for their time in reviewing our manuscript and providing feedback.

This article presents a user interface for the community hydrologic model GS-Flow using the community GIS package GRASS. This manuscript is well written and clearly presented. The interface is well documented. However, I am having trouble seeing the primary goal or take-home message for the readership of GMD. Is there a science or educational motivation for this work that allows users to do something they can't already do with the existing PRMS / Modflow approach? I like this manuscript and think it's well written but as currently framed, for me, misses this key point and reads much more like a user manual than a scientific article. I think revisions are needed to bring this critical point forward.

We are glad that the reviewer liked our manuscript and found it to be well-written. We also appreciate the reviewer bringing to our attention that we needed to clarify the scientific merit of the work, which we realize was very inadequately described in the original manuscript. Scientific understanding of integrated hydrologic processes has been stymied by the inaccessibility of complex models for many researchers and resource managers; the major advancement of our work is to provide a robust and flexible software for implementing the USGS's groundwater and surface-water flow model – GSFLOW – across diverse hydro(geo)logic settings; importantly, this required the development of new GRASS GIS extensions that overcome common obstacles in creating automated and reproducible surface and subsurface model domains for integrated hydrologic models. We now realize that many of these points were almost entirely missing from the original manuscript version, and we have substantially revised the manuscript to address this major shortcoming.

Our edited Introduction now emphasizes that existing software for integrated hydrologic models do not provide freely accessible toolkits that fully cover pre- to post-processing steps (p. 2, lines 22-30), and that GSFLOW-GRASS addresses that gap (p.2 lines 31-34, p. 3 lines 1-6). The original manuscript version documented the new GRASS GIS extensions, but admittedly, it did so much like a manual and provided almost no background on the challenges of creating robust and automated tools -- which have led to a general unavailability of such solutions predating our toolbox. A new paragraph has been added to the Introduction to present the technical advancements with these GRASS GIS extensions (p. 3 Lines 7-21). Further, we have entirely re-written Section 3.2 on the GRASS GIS domain builder (p. 10-12), so that it now explicitly describes what was implemented to solve specific known problems with stream network delineation. Finally, we also made major changes to Section 4 on the Examples, in order to explain how each model implementation demonstrates a different capability of the domain builder (specifically, p. 21 Lines 3-10 for Shullcas, p. 21 Line 32- p. 22 Line 4 for Santa Rosa, and p. 23 Line 10- p. 24 Line 4 for Cannon River). In particular, these examples demonstrate how GSFLOW-GRASS handles known challenges with various degrees of drainage integration, landscape relief, and grid resolution, as well as the presence of irregular coastal boundaries.

minor comments p1. lines 1-6. I think a better firs paragraph can help motivate this work's main takeaway point more clearly.

We have entirely rewritten the first paragraph (as well as most of the rest of the Introduction section) to highlight key motivations for "streamlined access to models that integrate surface and subsurface processes," which includes tools that address "challenges of of generating computationally robust surface and sub-surface model domains" (p. 1 Lines 1-8).

p1. line 9. GS-flow isn't an integrated model, it is coupled. Integrated models are defined to solve 3D richards' equation and the shallow water equations in an implicit framework to capture these coupled, nonlinear processes. This should be clarified in the revised manuscript.

We appreciate the reviewer's rigor in distinguishing between the use of the terms "integrated" and "coupled." Indeed, GSFLOW is a "coupled" model in that it that employs an iterative method to link the base codes of PRMS and MODFLOW. We chose to also refer to GSFLOW as an "integrated" model following the USGS's use of that term. The GSFLOW manual (Markstrom et al., 2008) distinguishes between two types of "integrated" models - "fully integrated" models that simultaneously solve surface and subsurface domain equations (what the reviewer calls "integrated") and "coupled regions" models that iterate between solutions for each set of equations (what the reviewer calls "coupled"). Thus, the USGS presents GSFLOW as an "integrated" model of the "coupled regions" type.

In order not to confuse model users, who will likely also be looking at the GSFLOW manual, we elected to adopt the same terminology as the USGS. However, we do now clarify that GSFLOW is not "fully integrated" but is instead "coupled" on p. 2 Lines 18-20. In fact, as a coupled model, GSFLOW still requires all the individual input files of both underlying models, which accounts for much of the laborious and time-consuming process of implementing GSFLOW. This motivation for a bundled toolkit solution for coupled models is now highlighted on p. 2 Lines 20-22.

p3. lines 7-11. Is this platform run in parallel? My understanding is not, nor is GS-flow parallel. I'm confused by this statement.

We realize from this comment that our original wording was confusing. No, GSFLOW-GRASS is not set up to run in parallel. The statement referenced by the reviewer was simply referring to the general advantages of using gridded domains, one of which is easier porting to parallel systems if desired. We edited the text to now read: "In general, gridded domains are easier to construct and extend to parallelized computational systems, and they allow flexible spatial specification of soil and land-cover heterogeneity." (p. 4 Lines 11-13).

p3. line 10. I think the comment about triangulated grids providing better water balance is unsubstantiated and perhaps false. Most triangulated formulations are not even locally mass conservative which leads to local water balance error. GS-flow also uses structured grinding, which seems contradictory to these statements.

We realized that we left out a critical detail – we meant to specify that TINs show better water balance performance IF they are implemented with the finite volume method (because the finite volume method is mass-conserving). We edited the text to clarify the water balance advantage with finite volume on p. 4 Line 15-16. GSFLOW-GRASS uses rectangular grid cells for the MODFLOW subsurface component, but it uses irregular sub-basin HRUs for the PRMS surface component. We consolidated all the information about the GSFLOW-GRASS domain discretization in the first paragraph of our Methods section to make this clearer (p. 9 Lines 2-7).

p3. line 24. again, GS-flow isn't integrated (or "integrated") and I don't know what 'integrated-coupled' even means.

See our above explanation of our use of "integrated," "fully integrated," and "coupled." However, we do realize that the wording mentioned by the reviewer was awkward and removed it on p. 3 Line 26.

p4. line 26+. This paragraph is short and confusing. Please reword.

We believe the reviewer was confused by the vagueness of "different modes" and the ambiguous "they" in the original text ("Table 1 in the GSFLOW manual (Markstrom et al., 2008) lists all PRMS modules, MODFLOW stress packages, and GSFLOW modules used by GSFLOW in the different modes. This section includes a brief description of the main processes they represent."). We re-wrote the paragraph more clearly as follows: "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.” (p. 5 Lines 2-5).*

p22. lines 7+. These don’t strike me as conclusions and read a bit like an advertisement. To my central point, what is the scientific motivation and conclusions reached by this work. Reworking this paragraph would help that substantially.

After reading this review, we agree that the Conclusion should be re-written and have now done so. The new Conclusions section emphasizes the technical advances provided by the GRASS GIS domain builder tools, the capabilities of GSFLOW-GRASS across diverse settings demonstrated by the model examples, and the value of this new toolkit for making integrated hydrologic modeling more accessible; we also end with a list of potential future extensions of this toolbox.

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

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

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

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

Abstract.

~~Water flow through catchments sustains ecosystems and human activity, shapes landscapes, and links climate to the outermost layers of the solid Earth. The profound~~ 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, a ~~straightforward~~ bundled set of open-source tools that develops inputs for~~and runs~~, executes, and graphically displays the results of GSFLOW, the U.S. Geological Survey's coupled ~~groundwater–surface-water~~ 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 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 ~~is written in Python as a set of (1) GRASS GIS extensions, (2) input-file-builder scripts, and (3) visualization scripts. We developed a set of custom GRASS GIS commands that generate "hydrologic response units" for surface water, discretized topologically as sub-basins of the tributary network; build the MODFLOW grid; and add necessary attributes to each of these geospatial units. These GIS outputs are interpreted by a second set of Python scripts, which link them to hydrologic variables, build inputs to GSFLOW, and run GSFLOW. Lastly, GSFLOW output files are used to produce figures and time-lapse movies of simulation results using a third set of post-processing Python scripts. We demonstrate the broad applicability of these tools to diverse settings through examples based on: the high Peruvian Andes, the Channel Islands of California, and the formerly-glaciated Upper Mississippi~~

~~valley in Minnesota .~~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~~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).

~~Although the USGS completed the software development work required to fuse MODFLOW and PRMS into a single model, the~~ The process of implementing GSFLOW includes many hurdles that require significant time and computational knowledge to overcome~~(Gardner et al., in review) . To run GSFLOW , the user must first generate multiple formatted ASCII files corresponding to the individual~~ . 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 ~~model inputs, along with~~ that requires nearly all the individual input files for each of the two original models as well as an additional GSFLOW-specific ~~file containing information about the alignment of these two single-domain models~~linkage file. This means that to run GSFLOW, the user bears the burden of first generating a multitude of formatted ASCII files while ensuring that the three sets of input files are consistent 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 ~~each of these models . However, with the exception of a single for-fee software product by Earthfx (http://www.earthfx.com/) , these tools neither help users conceptually link the different process domains represented in PRMS and MODFLOW nor generate the files that link them. The expertise required to develop inputs to GSFLOW are not often available at research institutes or management organizations, and there is a potential to significantly increase GSFLOW’s user base~~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., in review) , 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 ~~software for developing GSFLOW models, and support software for integrated models in general,~~ for developing integrated hydrologic models such as GSFLOW motivates our present work~~and we hope~~ , which we anticipate will enable more widespread hydrologic modeling.

~~Here we offer a free, platform-flexible, and easy-to-use utility tool that automatically builds all input files needed for GSFLOW. This tool consists of a domain-building module that generates and links a network of stream segments and watershed sub-basins to a MODFLOW grid; a set of Python scripts that create self-consistent GSFLOW, PRMS, and MODFLOW input files; and a final post-processing set of Python scripts for visualizing simulation results. The domain-building module requires a digital elevation model (DEM) as input , and uses GRASS GIS, an~~

Our overarching goal is to develop a bundled package – "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 ~~GIS platform to~~ 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 ~~as the surface-water (PRMS) discretization. GRASS GIS is computationally efficient and offers a wide range of raster and vector algorithms. Use of open-source~~ linked to subsurface grid cells. Open-source software facilitates implementation of ~~the~~ GSFLOW-GRASS by diverse academic, government, and individual entities, enables further community development of GSFLOW-GRASS, and aligns with the USGS's goal to make its resources ~~universally accessible. Our effort complements ongoing USGS development of an ArcGIS-based input file creator that employs overlapping rectangular grids for both MODFLOW and PRMS (Gardner et al., in review) , in contrast to topologically based surface units created in GSFLOW-GRASS.~~ 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. While this method is mathematically correct, its 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. It is therefore also possible to edit drainage structure by hand. The main technical advancement of 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 to coastal watersheds. These new computational capabilities enable rapid, automated delineation of surface-water drainage networks linked to subsurface domains across any generalized landscape and practical resolution. By doing this all within a framework that also includes open-source model input and post-processing tools, GSFLOW-GRASS presents a solution toward more accessible integrated hydrologic modeling.

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

2 Background

2.1 GSFLOW

GSFLOW simulates spatially distributed surface to subsurface water flow in a watershed using modified model codes from PRMS and MODFLOW. Although GSFLOW can run in modes equivalent to the stand-alone PRMS-IV model and the stand-alone MODFLOW model, only the "integrated" ~~coupled~~ version is described here. Near-surface watershed processes within the shallow "soil zone," including evapotranspiration, infiltration, runoff, and interflow, are represented by the PRMS sub-component of GSFLOW. Groundwater flow below the "soil zone," including vertical soil water movement in the deeper unsaturated zone and saturated flow through horizontal aquifer layers, is represented by ~~MODFLOW~~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) documents 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 ~~, of any shape~~ (Figure 1B). These are used for calculations of the upper soil zone and the part of the surface not covered by the stream network. The MODFLOW sub-component uses rectangular grid cells for the deeper subsurface (Figure 1C) and to further discretize the stream network into reaches (Figure 1D). Establishing reaches as the fundamental unit of computation for the stream network instead of segments makes it possible to resolve fine spatial resolution groundwater-surface exchanges. Like MODFLOW grid cells, HRUs can be set to rectangles, but they are also commonly defined topologically to correspond to sub-basins, as they are in our approach (Figure 1). ~~An extension to this could include further subdividing~~ 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 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). Other hydrological models with ungridded domains use topographically defined subbasins ~~according to vegetation, soil type, or other geographic features to produce HRUs.~~ 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~~ 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.

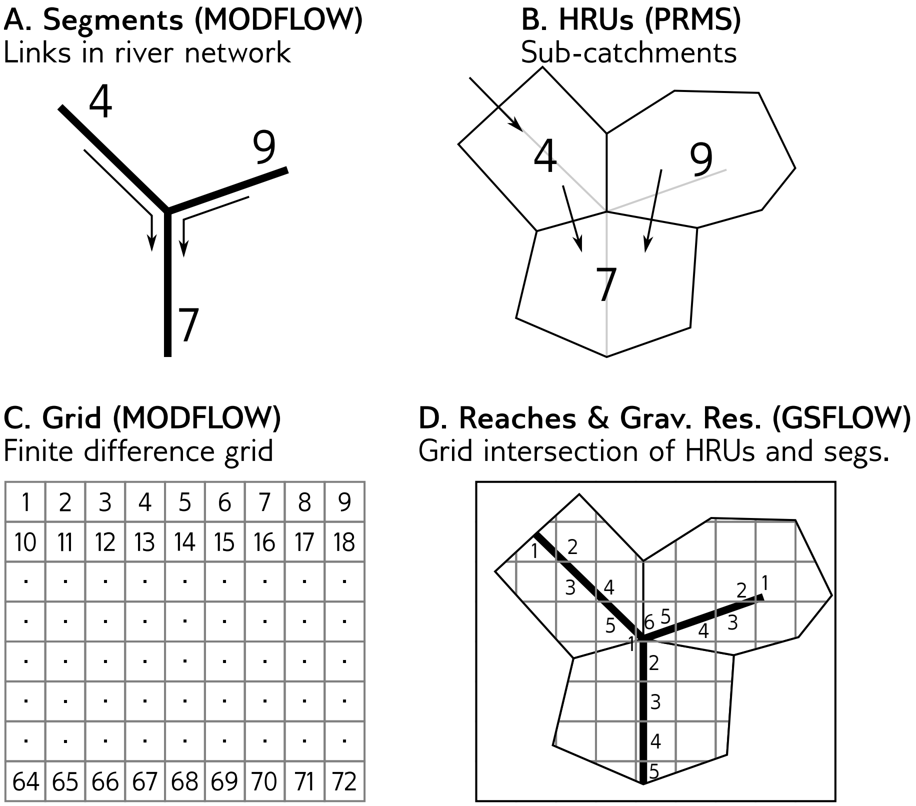


Figure 1. Major features of the GSFLOW geometry. **A.** Each segment is one link in the network. At each node, two tributary segments combine to flow into a single segment. Each is numbered~~, but~~. They need not be in any particular order, as indicated, but a downstream-increasing numbering scheme 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

~~Table 1~~ 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)~~lists all~~. 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 ~~GSFLOW modules used by GSFLOWin the different modes. This section includes a brief description of the main processes they represent.~~ model coupling procedures captured by GSFLOW.

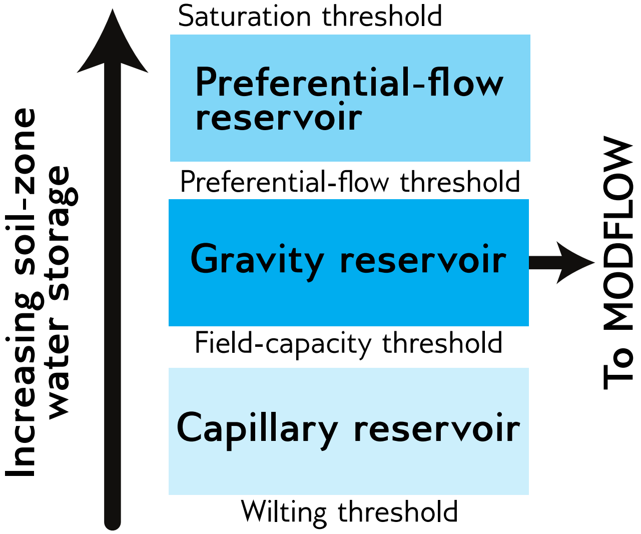


Figure 2. (Adapted from Markstrom et al. (2008), Figure 12) Soil water storage reservoirs in the PRMS component of GSFLOW. Within each HRU, soil water accounting calculations are carried out for three conceptual reservoirs in the order of increasing water storage and according to user-specified parameters.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

.

The PRMS component of GSFLOW includes ~~various~~ modules that can convert commonly available climate data into complete forcing inputs needed for model simulations. These ~~modules include different~~ include methods for determining potential solar radiation, potential evapotranspiration, and snow accumulation ~~/~~ or depletion; they also include different ~~schemes~~ 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 ~~efficient~~ 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~~ field capacity (parameter *soil_moist_max*), water transfers from the capillary zone to “gravity reservoirs~~,”~~”, where water can flow horizontally as slow interflow (based on parameters *slowcoef_lin* and *slowcoef_sq*) ~~and~~ or drain vertically into the deeper subsurface domain that is handled by MODFLOW (based on parameters *ssr2gw_rate*, *ssr2gw_exp*, and *ssrmax_coef*)~~; gravity~~ . 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

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

as Dunnian (saturation-excess) runoff. Hortonian (infiltration-excess) runoff calculations apply for impervious fractions of HRUs (set by parameter *hru_percent_imperv*). Surface runoff and interflow are routed between HRUs, using a cascading flow scheme that follows user-specified indexing of linked HRUs, and eventually reaches the stream network. ~~A cascading HRU network is not needed for our domain formulation, which comprises sub-basin HRUs that route water directly to stream segments (Section 3.2) .~~

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

Streamflow, as calculated by the MODFLOW component, includes inputs from upstream reaches, surface runoff and interflow from the PRMS component, base flow from the saturated zone discharge, and flows from possible underlying unsaturated areas. Outputs include flow to downstream reaches, leakage to groundwater, and flows to possible underlying unsaturated areas. Discharge across the streambed follows Darcy~~calculations~~ 's law with specified streambed hydraulic properties. ~~User-specified~~

~~additional inputs and outputs are also allowed.~~ Five different options exist for stream discharge and head computations (parameter *ICALC*). The user can specify stream depths for each reach; apply Manning's equation to an assumed wide rectangular channel; apply Manning's equation for an eight-point-based channel and floodplain geometry; apply at-a-station power-law relationships between discharge, flow width, and flow depth (Leopold and Maddock, 1953); or specify an input look-up table of hydraulic geometries for each segment. Streamflow can be simulated as either steady-state flow (parameter *IRTFLG* = 0), where outflow to the next stream reach balances inputs, or as transient flow (parameter *IRTFLG* > 0), using a kinematic wave formulation for ~~surface water~~ surface-water routing in channels, which applies the assumption that the water surface slope approximates the friction slope, and therefore negates backwater effects.

~~It should be noted that some~~ Some modifications were made to the original stand-alone PRMS and MODFLOW codes for their use in GSFLOW. Notably, the soil-zone structure of PRMS ~~is~~ was significantly altered to facilitate ~~linking to~~ its linkage with a MODFLOW subsurface domain. Other modifications are noted in the GSFLOW manual ~~(see sections on “Changes to PRMS” and “Changes to MODFLOW-2005”)~~(Markstrom et al., 2008, see sections on “Changes to PRMS” and “Changes to MODFLOW- An additional feature starting in version 1.2.0 that is not described in the original manual is the inclusion of MODFLOW-NWT (Niswonger et al., 2011), a more numerically robust update to MODFLOW-2005 for groundwater flow.

2.2 GRASS GIS

GRASS GIS is an open-source, multi-purpose, and cross-platform geographic information system (Neteler and Mitasova, 2008; Neteler et al., 2008, 2012) that supports utilities for efficient raster and vector computations (Shapiro and Westervelt, 1994; Mitasova et al., 1995; 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 utilities are written for the ~~current~~ most recent stable release version of GRASS GIS, v7.4. This supports Python scripting for both high-level built-in commands and for low-level access to database entries and vector geometries (Zambelli et al., 2013). We take advantage of these capabilities to develop an automated workflow to build GSFLOW inputs through GRASS GIS.

We chose GRASS GIS as the interface to develop inputs because (1) it is open-source and cross-platform; (2) it enforces rigid vector topology, ~~and it~~ which is critical for building stream networks; (3) its generic Python scripting library and PyGRASS Application Programming Interface (API) make it easy to develop new extensions; (4) these extensions may be added to the official subversion (svn) repository, from which they can be automatically downloaded and installed on users' computers using the **g.extension** command~~,~~ ; and (5) it ~~supplies~~ provides a GUI and command-line interface (CLI) that are consistent with one another. ~~These~~ The GUI and CLI interfaces are not required for GSFLOW-GRASS, because the GRASS GIS component is handled mostly behind-the-scenes by a ~~batch processing~~ 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 extensions as building blocks. The open-source aspect ~~is~~ 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.

3 Methods

We adopt a heterogeneous surface and subsurface computational domain for 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 sub-basins rather than a gridded surface domain, 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 gradient-based), numerical differences between sub-basin and gridded HRU's are difficult to predict, but the automated GSFLOW-GRASS toolbox can help enable future testing to rigorously interrogate their respective performances.

GSFLOW-GRASS strikes a balance between generating a ready-to-go GSFLOW implementation and providing flexibility to customize applications. ~~Out-of-the-box, and~~ With a newly developed set of automated and robust GIS domain builder tools, GSFLOW-GRASS can be applied to any DEM to produce GSFLOW model simulations, with only a few steps to set up the model with the user's computer directory system~~, the toolkit can be applied to any DEM to readily produce GSFLOW model simulations for a watershed. Then, for further tuning~~. For further model-tuning, all scripts in the toolbox are open-source and commented to allow changes to any parameter and development of optional GSFLOW capabilities not included in the default GSFLOW-GRASS implementation. ~~While many~~ Many popular hydrologic model implementation programs have GUIs~~(e.g.,~~ , 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)~~), which~~ . While these are easiest for novice model users, GUIs can be challenging to develop for cross-platform implementations ~~, are often unstable,~~ and generally support less flexibility for customization. Thus, we chose a mostly command-line ~~GRASS-GIS / Python program-based approach. GSFLOW-GRASS~~ approach, which has been designed and tested for use on Linux and Windows operating systems.

3.1 User-specified ~~Settings~~ settings and ~~Model Inputs~~model inputs

~~For a seamless execution of~~

To seamlessly unify the different GSFLOW-GRASS ~~, a Settings text file is used for specifying all inputs needed, including those for the~~ functionalities, including the automated GRASS GIS domain builder, GSFLOW input-file builder, and visualization components~~described below. We note, however, that only certain model inputs may be set through~~ , 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, this toolkit applies default values for most inputs, and only includes~~ ease of use, only a handful of application-specific and commonly adjusted inputs ~~in the Settings text file. When using the default implementation, the user only modifies the Settings text fileand does not need to directly handle any GRASS GIS tools or Python scripts. However, for flexibility, additional parameters may be changed in~~ may be assigned using the Settings file, and default parameter values are applied elsewhere. While the default (and simplest) approach to GSFLOW-GRASS is to modify only the ~~open-source scripts~~ 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 ~~specified~~ defined in the GSFLOW manual ~~, including those mentioned in Section 2.1.2 . All inputsfrom the Settings file are read in and processed by the **ReadSettings.py** script.~~ 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.

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** 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 data structures. Other software 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., in review) . 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 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 first section of the~~

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

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

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 ~~specified infiltration to the MODFLOW domain~~ a specified infiltration rate (see below) to calculate reasonable ~~conditions for initializing a~~ initial groundwater head conditions for the subsequent transient simulation ~~within the spin-up run. The~~ that includes both the surface and subsurface domains; the steady-state step can be essential for obtaining ~~initial conditions for the groundwater system for numerically convergent MODFLOW results and helps with reaching more realistic transient state solutions~~ 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 “domain~~The “time” section is used to specify the temporal window ~~and the vertical discretization~~of the simulation. The “climate inputs” section sets input parameters for the PRMS “climate_hru” option, which is ~~not generated by the GRASS module. GSFLOW-GRASS allows for multiple vertical layers, each of which can have a different horizontally uniform thickness, with the top and bottom of each layer mirroring land surface topography. This section is also used to specify the start date and end date of~~ the ~~simulation.~~

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

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 (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 different geometries and resolutions for surface-water HRUs and the groundwater-flow grid.

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 network will be topologically correct, it may 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, its elevation values are the 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 ~~built ten~~ created eleven new GRASS GIS “extensions”, also called “add-ons”, ~~which are commands~~ that work alongside ~~existing GRASS GIS functionality~~ core GRASS GIS commands to transform user ~~specifications and a single digital elevation model (~~inputs (including a single DEM) into a set of GSFLOW inputs ~~. These~~ 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 ~~once~~ only one time using GRASS GIS, and then perform multiple runs for parameter ~~estimation~~ calibration or scenario tests without having to repeat the domain construction. This 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’s Python input-builder scripts.

~~The toolkit’s GRASS GIS extensions define HRUs topologically – that is, based on the drainage network. This follows both the natural discretization of the landscape and the architecture of PRMS, which was developed to support flow routing through~~

~~sub-catchments (Markstrom et al., 2015) . This approach is complementary to the grid-cell HRU approach of (Gardner et al., in review) .~~

In the first step of the ~~GRASS~~ fully automated domain-building workflow, GRASS GIS imports a user-provided DEM ~~, removes "off-map" cells that may have flow contributions from outside of the DEM, hydrologically corrects the DEM elevations, and builds a drainage network (Metz et al., 2011) . We then construct a Hortonion drainage network and the sub-catchments~~ 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 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) . Sub-basins associated with each ~~segment in that network using the **r.stream.*** set of extensions by Jasiewicz and Metz (2011) . In creating this network, the user must supply~~ 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 ~~for a stream segment to be defined; this is done with the help of our **r.cell.area** command, which automatically creates a raster of grid cell sizes. Smaller drainage area thresholds generate a larger set of HRUs, and the largest number of HRUs that can be defined is a function of the total drainage basin area andthe resolution of the DEM.~~ 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 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 a unique positive integer identifier, called a "category" in GRASS GIS. For each segment in the drainage network, ~~v.stream.network~~ **v.stream.network** writes the category value of the immediately downstream stream segment to the "tostream" column in its associated attribute table row. Any stream segment exiting the map area is given a "tostream" value of 0. At this point, the user may make edits to the structure of the drainage basin, for example, by correcting stream courses to align with human-developed drainage structures. After this, ~~we limit~~ the study area is limited to a single drainage basin using ~~our~~ the new **v.stream.inbasin** extension~~. At this point,~~ , completing the development of the drainage network geometry and topology~~is developed, and we move on to primarily using our new **r/v.gsflow.*** set of extensions to populate attribute values and link the topographically driven surface-water flow geometry with a regular subsurface grid for MODFLOW.~~ .

~~The final step within the GRASS workflow uses a sequence of new commands, designated by "gsflow," and associated built-in GRASS GIS utilities to provide the geometry, topology, and much of the stream parameters needed to build GSFLOW input files (Section 3.3).~~

Each segment is then supplied with attributes through the **v.gsflow.segments** extension. This numbers each river channel segment for GSFLOW (Figure 1A) and ~~sets the attributes that define its~~ populates the associated database table with hydraulic geometry, channel roughness ~~, input discharge beyond that from precipitation~~ (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. ~~In its current form, **v.gsflow.segments** applies Manning's equation to a wide channel (*ICALC*= 1) and allows the user to set a single channel width and Manning's n (in-channel roughness coefficient for flow resistance)across the whole domain. This is designed to be a starting point for simulations that is not necessarily realistic; field data on channel geometries come in a variety of forms, which would be difficult to accommodate in a generalized tool. Instead, we encourage users to modify the GRASS GIS database tables. This can be performed via **v.db.update** within GRASS GIS; by modifying the GRASS GIS extensions that we have built to enable additional user-set options that specify hydraulic geometry; and /or to edit the exported files, **segments.txt** and **segments.txt** in the **GIS** subfolder. **v.gsflow.hruparams** creates and populates attributes of each HRU (Figure 1B)~~

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 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. **v.gsflow.grid** ~~aligns and~~ builds the MODFLOW grid ~~along the higher-resolution (i.e. original flow-routing) DEM cell edges~~ (Figure 1C) ~~; it also identifies the grid cell directly downstream of the drainage outlet, which can be used to generate a constant-head boundary condition for improved simulation performance (see Section 3.3.3)~~while enforcing that it must contain only whole DEM grid cells, and that its edges must align with cell edges in the DEM. **r.gsflow.hydrodem** ~~generates a hydrologically corrected DEM at the resolution~~ then hydrologically corrects the elevations of the MODFLOW grid ~~, which the user can make coarser than the flow-routing DEM for computational efficiency. In order to maintain downstream flow while downsampling, the lowest-elevation pixel that corresponds to each MODFLOW stream cell is selected as its elevation ,~~ cells; cells that contain stream segments are given a surface elevation corresponding to the ~~stream channel within these cells, while non-stream~~ lowest-elevation overlapping pixel on the flow-routing DEM, while all other MODFLOW cells are assigned the mean elevation from the ~~higher-resolution~~ corresponding cells in the flow-routing DEM. **v.gsflow.reaches** and **v.gsflow.gravres** construct the ~~“reaches ” and “gravity reservoirs ”~~reaches and gravity reservoirs (Section 3.1), which are the intersection of segments and HRUs, respectively, with each MODFLOW grid cell (Figure 1D). The ~~reaches include~~ database table for the reaches includes values for the thickness of the ~~stream bed~~ stream-bed sediment (defaults to 1 ~~meter~~m) and its hydraulic conductivity (defaults to 5 m/d, characteristic of sand and gravel). ~~Default values are set by~~ GSFLOW supports more input options than we have defined for our GRASS GIS ~~**buildDomainGRASS**~~**v**~~**.py, though the user can modify these**~~**gsflow.*** commands, but we have included the most common options, as well as the **v.gsflow.mapdata** tool for users to add other attributes to database tables.

Finally, the generated attributes and geometries are exported. **buildDomainGRASS.py** ~~then exports the rasters, which comprise a~~ exports a rasterized “basin mask” (1 in the basin, 0 outside) and the hydrologically corrected DEM ~~, both at the resolution of the MODFLOW cells, using the built-in commands **g.region** and **r.out.ascii**. After this, **v.gsflow.output** creates comma-separated variables (CSV) files for the segments, HRUs, reaches, gravity reservoirs, the pour point at the outlet of the basin, and constant-head boundary condition cells. Finally, **buildDomainGRASS.py** exports the derived vector maps in shapefile formatusing the built-in GRASS GIS **v.out.ogr** command; these include~~ at the MODFLOW grid resolution, as well as vectorized GIS data (shapefile format) for the HRUs, gravity reservoirs, MODFLOW grid, ~~full study basin area,~~ stream segments, stream reaches, pour point, full study basin area, and downstream boundary-condition cells. ~~All of these outputs are written to the **GIS** subdirectory of the project folder, whose location is specified in the Settings file. GSFLOW supports more input options than we have defined for our GRASS GIS **v.gsflow.*** commands, but we have included the most common options, and the structure of the open-source code is straightforward to modify to include additional attribute values.~~ **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.

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

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

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

3.3 GSFLOW Input File Builder

~~With~~

GSFLOW-GRASS includes a set of input-file builder scripts that are streamlined to incorporate the model domain ~~constructed using~~ discretization constructed by the GRASS GIS workflow ~~, GSFLOW input files can be created with GSFLOW-GRASS's builder scripts~~ and generate corresponding model inputs for the GSFLOW control file, PRMS-type input files, and MODFLOW-type input files. Most of the new features ~~unique to GSFLOW (~~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 ~~the~~ use of a "control file" as the main interface file, ~~the use of~~ "modules" for different computational options, and the PRMS input file syntax. In contrast, MODFLOW uses a "name file" as its main interface file, implements "packages" for computational options, and follows its own file syntax. ~~Execution of the~~ The following builder scripts ~~is automated~~ handle these different

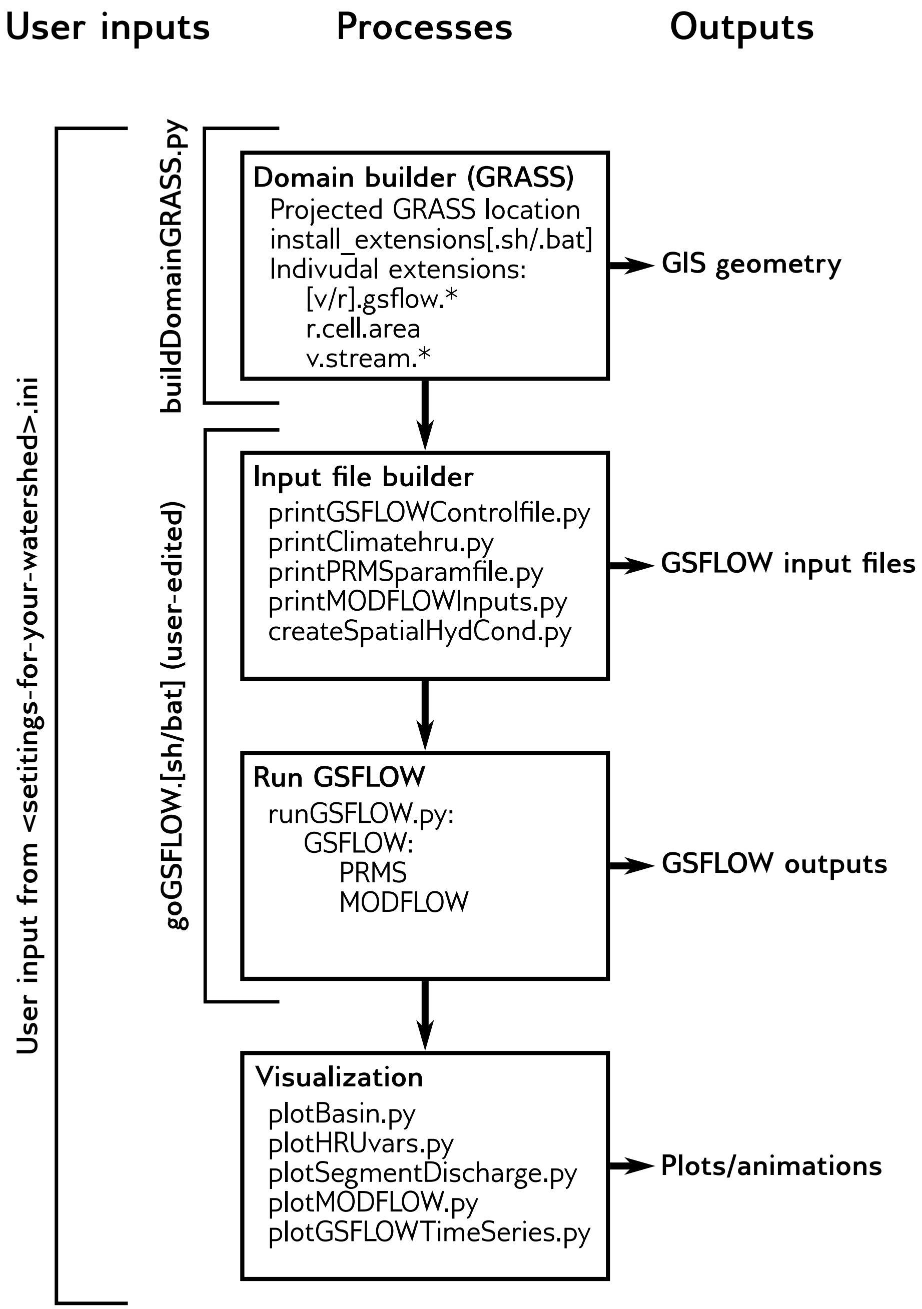


Figure 3. GSFLOW-GRASS workflow. The user: (1) creates a ***.ini** file based on their study catchment; (2) creates a projected GRASS GIS location; (3) runs **buildDomainGRASS.py**; (4) edits and runs **goGSFLOW.py**. After this, they may use GSFLOW-GRASS's visualization tools to study the GIS and model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

.

formats and are automatically executed through the toolkit's Run file (Section 3.4)~~, but the~~ . The builder scripts may also be customized for extensions beyond the default implementation.

3.3.1 GSFLOW ~~Control File~~control file

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

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

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

After the six parameter input sections in **printGSFLOWControlfile.py**, the script builds the control file and then generates an executable file (shell script for Linux ~~and~~ or batch file for Windows) for running GSFLOW with the control file. After all other input files are created, this executable ~~gets~~ 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 toolkit.

3.3.2 PRMS-type ~~Input Files~~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~~includes~~, the script **printClimatehru.py** ~~, which~~ takes daily observations from a single file and distributes them uniformly over all HRUs. The toolkit handles the minimum required climate variables – daily precipitation, maximum temperature, and minimum temperature, and it is set up to be readily extended to also include humidity, solar radiation, and/or wind speed. ~~This spatially-uniform approach is appropriate~~ A spatially uniform approach may be acceptable where the size of a rainstorm is ~~significantly~~ typically greater than the size of a catchment and climatic variables vary only weakly with slope and aspect~~; larger~~. Larger and higher-relief catchments require ~~spatially-distributed~~ spatially distributed climate inputs for realistic outputs~~, and~~; these require custom inputs from the ~~end user~~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 first includes a section~~ The script includes sections for domain dimensions ~~, including the list variable *dim* for the different dimension names and *dim* for their corresponding values, and then a section for parameters , including the list variable *par* for the different parameter names and *par* for their corresponding values. Both sections include various entries that~~ 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, ~~it~~ the parameter files requires a substantial number of ~~parameters~~ 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~~input files

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

The script **printMODFLOWInputs.py** calls the functions from **MODFLOWLib.py** to create a set of internally consistent input files that incorporate the domains constructed by the GRASS-GIS workflow (Section 3.2) and conform to the simulation directory structure established through ~~our~~ 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 consistent directory structure with the rest of the input files and the visualization scripts.

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

~~These may be modified by adjusting the interface to the GRASS GIS modules (see Section 3.2 and the accompanying user's guide)~~*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~~**run file**

For the user's convenience, the 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**~~) that uses~~ . The Run file collects input from a specified Settings file and then runs all of the above input-file builder scripts~~as well as~~ ; the script **runGSFLOW.py**, which launches the GSFLOW simulation~~. Thus, as~~ ; 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 ~~change the default implementation beyond features~~ use more features than are exposed in the Settings file, no direct interface with the ~~Python program or~~ code is required to run GSFLOW-GRASS. This ~~allows for~~ permits a "quick-start" implementations of GSFLOW, which can substantially lower the barrier to entry for using this model.

The Run file ~~can be implemented~~ may be implemented only after the model domain is generated through **buildDomainGRASS.py**. The ~~toolkit separates out the GRASS domain builder~~ 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 ~~want to re-try simulations with various parameter inputs~~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~~**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)~~ (e.g., QGIS: QGIS Development Team, 2013) outside of GSFLOW-GRASS.

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

For basin-total GSFLOW results, the Python script **plotGSFLOWTimeSeries.py** generates time series lines for user-selected variables from the main GSFLOW ~~csv~~ 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 ~~body~~ 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 ~~the~~ 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 ~~may be~~ 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 extension ".bhd," but this does require manual post-processing steps. For Model Viewer, the user needs to copy all MODFLOW input and output files to a new folder inside the Model Viewer project directory and select the namefile when prompted. For Model Muse, the user must first delete the line that starts with "IWRT" from the name file in order to load the project into the program. Once the project settings are loaded into ModelMuse, the user can use the "import model results" tool to select the binary head file.

4 Examples

~~As examples of the application~~ Three implementations of GSFLOW-GRASS serve the dual purposes of demonstrating (1) the robustness of the newly developed GRASS GIS domain builder across diverse topographic settings, including those prone to problems with standard GIS stream network tools, and (2) the variety of hydrological processes that can be assessed with the use of GSFLOW-GRASS~~, we have chosen~~. The implementations utilize topographic and climatic inputs from widely differing

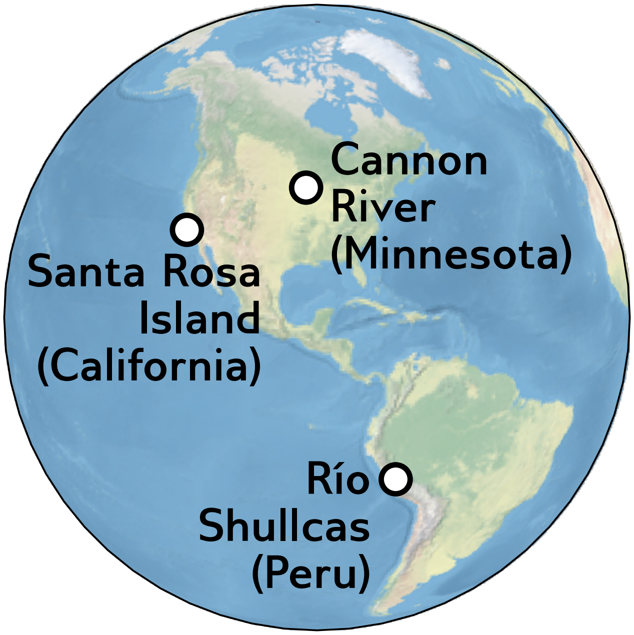


Figure 4. Our test sites ~~span~~ include the high Andes, a mountainous island, and a ~~formerly-glaciated~~ formerly glaciated Mississippi tributary.

sites to show the range of applicability of the new toolbox. It is important to note, however, that no calibration effort was made to match field observations. 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, which has undergone land-cover change impacts; and the ~~formerly-glaciated Cannon River , which~~ formerly glaciated Cannon River watershed, in which water flows from intensely farmed uplands into an incised bedrock valley in Minnesota, USA (Figure 4). All regions contain complex hydrology with interactions between surface water and groundwater and are exemplars of practical management concerns. Together they span a range of environments: high to low elevations, steep to low-gradient catchments, coastal to inland settings, ~~tectonically-active~~ tectonically active to cratonal, and ~~glacier-influenced to temperate~~with partially- to fully- integrated drainage. They are impacted by modern climate and land-use change impacts on glaciers and agricultural (water and soil) resources (Shullcas) (Gómez et al., 2014; Arroyo Aliaga et al., 2015; Travezan Adauto, 2015), grazing-induced erosion (Santa Rosa) (Schumann et al., 2016) , and agricultural runoff and fertilizers (Cannon River) (Kreiling and Houser, 2016)~~, and grazing-induced erosion (Santa Rosa) (Minnich, 1982)~~ . Our choice of an example in the Peruvian Andes demonstrates how our entirely open-source modeling system may be applied to problems in the developing world~~.~~ , where computational resources may be limited for local environmental researchers and practitioners.

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

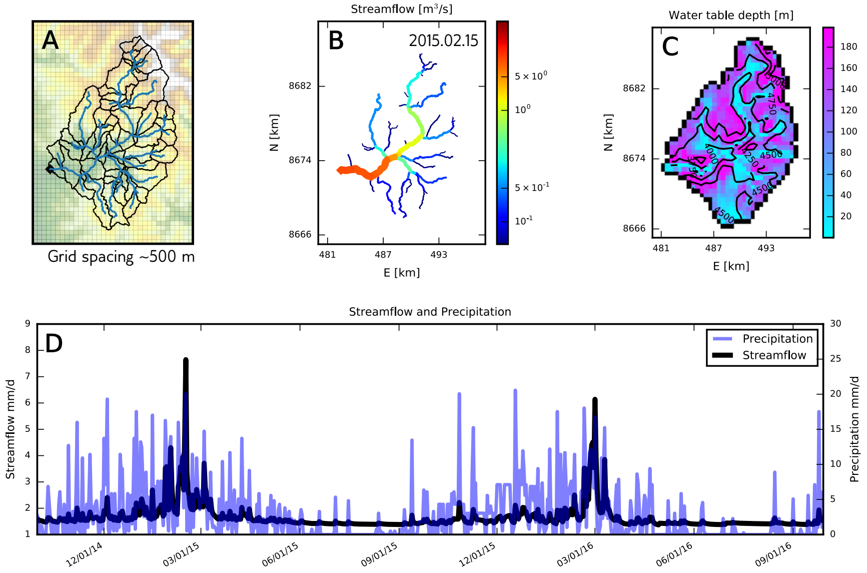


Figure 5. Model based on Río Shullcas, Peru. **(A)** Map with MODFLOW grid, HRU outlines, stream segments (blue), and digital elevation model. **(B)** ~~Streamflow accumulates~~ The model shows streamflow accumulating through ~~this~~ the mountainous drainage network. **(C)** ~~Groundwater is~~ Simulated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

tions in a variety of topographies and hydroclimatic conditions. ~~All three of these model exercises are purely schematic: no efforts were made to calibrate GSFLOW to field conditions. Figures 5–7 display sample inputs and outputs of the model simulations using~~ Preliminary simulations with the default 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 toolbox can be customized to quickly generate additional model runs to expedite the parameter calibration. Also, the GSFLOW-GRASS ~~toolkit for the three test sites~~toolbox can 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~~watershed, Peru

The first test case is ~~from~~ 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 Glaciers, which supply meltwater to the Shullcas River, are rapidly retreating (López-Moreno et al., 2014), causing concern over future water resources. However, a large proportion of the dry season stream discharge is composed of groundwater, driving the need to better understand groundwater-surface water interactions in the catchment. The steep topography and seasonal precipitation make the Shullcas River Watershed an apt testbed for examining the ability of ~~GSFLOW–GRASS~~ 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~~was used to prepare a simple hydrological model based on the Shullcas watershed. The modeled portion of the watershed has~~ 's use of topographically based irregular HRUs makes it possible to compute flow paths using high-resolution topography but with the fewest possible computational cells corresponding to fundamental surface-water hydrologic units – stream segments and sub-basins. These irregular drainage units are then linked to a relatively coarse regular groundwater (MODFLOW) grid whose cell elevations 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 170 km^2 and ranges in elevation from 3600 to 5500 m above sea level (a.s.l.). Using the GRASS domain-builder, the watershed was divided into 59 ~~sub-catchment~~ sub-basin HRUs based on an ASTER 30 m resolution DEM (Tachikawa et al., 2011). The subsurface was represented by a single 200 m thick MODFLOW layer, with a horizontal discretization of 46 rows, each with a length of 485 m, by 33 columns, each with a width of 492 m. ~~After an initial steady-state stress period in the "spin-up" run mode, the simulation ran for approximately three years from August 26, 2013 to September 29, 2016. Meteorological data – including maximum temperature, minimum temperature, and precipitation – were taken~~ Meteorological data were obtained from the Peruvian Meteorological Office (SENMAHI) online database~~, and are used to drive the model. Our~~ .

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 variability. ~~For all examples, results are shown after the effects of the preliminary steady-state conditions become minimal.~~ The GSFLOW-GRASS post-processing visualization tools were used to depict the spatial distribution of water table depths relative to the drainage network (Figure 5B-C), as well as time series of watershed forcing and responses (Figure 5D). Results shows that groundwater converging at the stream network could support baseflow contributions at the discharge outlet over time to sustain discharge during dry periods.

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 ~~simulated~~ hypothetical climate regimes that are wet, dry, and of average wetness when compared to

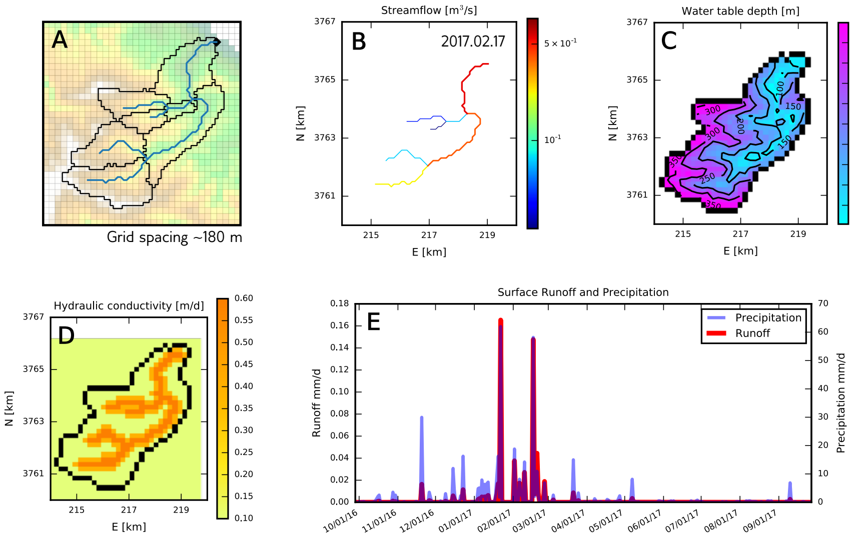


Figure 6. Model based on Water Canyon, Santa Rosa Island, California, USA. **(A)** Map with MODFLOW grid, HRU outlines, stream segments (blue), and digital elevation model. **(B)** Streamflow through the drainage network is ~~typically~~ simulated to be mostly less than 1 m^3/s in this semi-arid system.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

modern conditions. 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.

Here we apply the ~~GRASS-GSFLOW toolkit~~ GSFLOW-GRASS toolbox to model Water Canyon, one of the island's many drainages~~. The topographic relief in this example is intermediate between the Shullcas and Cannon River watersheds, and it~~ , 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 ~~our three~~ the three example sites in that its outflow drains to the ocean~~and that its semiarid climate leads to losing streams that may run dry (Jazwa et al., 2016)~~ , requiring that GSFLOW-GRASS appropriately handles NULL values for ocean grid cells. We drove this hydrologic model with topography derived from a 3 arcsecond SRTM DEM (Farr et al., 2007) ~~and~~ projected to a UTM coordinate system at 90 m resolution. This ~~DEM, which was the basis for surface-water routing calculations in the GRASS GIS module, is the coarsest of those in any of our examples. Based on user-provided inputs, these cells were downsampled by a factor of two to 180-meter resolution to develop~~ resolution is already low compared to the size of the island drainages, but we further down-sampled the DEM to

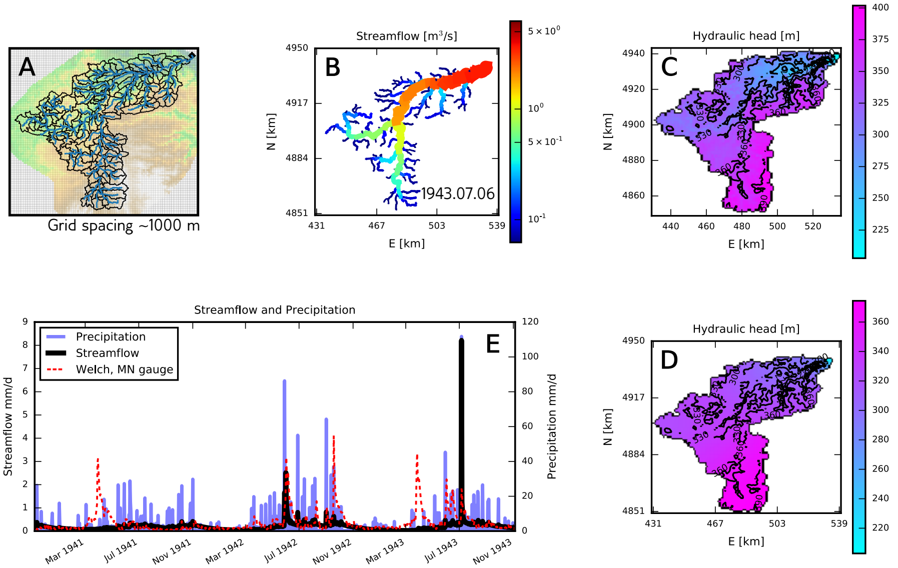


Figure 7. Model based on Cannon River, Minnesota, USA. **(A)** Map with MODFLOW grid, HRU outlines, stream segments (blue), and digital elevation model. **(B)** ~~Discharge on the Cannon River~~ Simulated discharge after an 11 cm rainfall event. **(C, D)** Two MODFLOW layers were implemented to represent an upper glacial till unit and the underlying fractured carbonate bedrock with likely higher hydraulic conductivity. ~~This generally~~ In the model, this low-relief catchment exhibits a shallow water table, except around the river gorge near the outlet. **(E)** The two-year hydrograph shows ~~the major 1943 storm event and its associated~~ uncalibrated discharge ~~, as~~ simulations matching observations reasonably well ~~as~~ during non-peak flood times but failing to capture many of the ~~two years prior~~actual peaks with default model parameters.

180 m resolution for the MODFLOW grid ~~; the degree of downsampling is chosen for computational efficiency, which is of little concern in this small and~~ to test the performance of our toolbox with a coarse-resolution ~~catchment. Basic weather data (rainfall, temperature, and humidity) were obtained~~ representation of a steep catchment.

GSFLOW-GRASS successfully produced simulation results shown in Figure 6 using weather data from the Western Regional Climate Center (wrcc.dri.edu) ~~, converted from their raw hourly frequency to the required daily inputs. This simple example in Figure 6 demonstrates~~ and spatially heterogeneous hydraulic conductivity (Figure 6D) generated with the example model input script. The semi-arid climate can lead to losing streams that may run dry (Jazwa et al., 2016) ; the GSFLOW-GRASS ~~'s capabilities in representing heterogeneous aquifer properties and probing surface hydrologic controls on erosion .~~ implementation correspondingly shows low streamflow in simulations (Figure 6B) but avoids numerical convergence problems that can arise when MODFLOW cells run dry. Figure 6E demonstrates how the post-processing tools can be used to evaluate surface runoff, a concern because of its potential for causing erosion on the island (Schumann et al., 2016) .

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 ~~use,~~ uses and its transient geomorphology (low-gradient headwaters above a high-gradient river) are common in the ~~formerly-glaciated~~ formerly glaciated Upper Midwest (Blumentritt et al., 2009; Carson et al., 2017). This leads to a suite of management concerns related to agricultural nutrients and fine sediments, and their interactions with both the surface water and the bedrock groundwater systems that underlie them (Tipping, 2006; Steenberg et al., 2013), thus motivating the need for integrated hydrologic modeling tools.

The Cannon River watershed covers 3720 km^2, with only 215 m of total relief, making it the largest and lowest-gradient of these study ~~areas. Its average elevation is 340 m above sea level. For its surface topography, we use~~ cases, leading to very different computational challenges than those in the steep watersheds. Its deglacial topography has not yet been organized by fluvial erosion into a linked valley network. In such settings, simple downslope flow-routing algorithms typically fail, and "pit filling" can produce spurious results by inappropriately modifying the real topography. GSFLOW-GRASS routes surface-water flow using the GRASS GIS **r.watershed** 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. Its success in this example demonstrates the ability of GSFLOW-GRASS to automatically create a topologically correct and linked drainage network in low-relief settings for hydrologic model simulations.

We implemented GSFLOW-GRASS using the Minnesota state-wide 1 m LiDAR data set (http://www.mngeo.state.mn.us/chouse/elevation/lidar.html), which we resampled to 15 m resolution. We ~~discretize~~ discretized the subsurface of the Cannon River watershed into 1 km MODFLOW grid cells~~, and define new stream segments and HRUs for each tributary that has a drainage area exceeding 9 km^2~~. Meteorological data from nearby Zumbrota, Minnesota was obtained from the Midwestern Regional Climate Center (Adresen et al., 2014). The northern, mid-continental setting makes the Cannon River watershed the example with the most seasonally distributed precipitation and strongest seasonal temperature differences. ~~GSFLOW outputs~~ 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. 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 toolkit, results such as those shown in Figure 7 can be used to evaluate ~~the susceptibility of~~ infiltration from overlying agricultural plots to shallow and low-gradient water tables~~to overlying agricultural inputs, which can also be episodically flushed into~~, as well as subsequent flushing of impacted shallow groundwater into the river channels during major storms.

5 Conclusions

~~We~~ 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, ~~an easy-to-use toolkit for implementing the integrated~~ a bundled toolkit for the coupled surface-water and groundwater model GSFLOW, using open-source Python scripts and GRASS GIS commands. ~~This allow~~ 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. ~~This tool improves~~ 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 designed GSFLOW-GRASS to strike a balance between functioning quickly out-of-the-box and including full flexibility for customizing model runs. A default implementation can be launched with no programming required by the user, while all toolkit scripts are commented with instructions for adjusting any model inputs if desired. Other key features of the toolkit include the use of all open-source software, enabling users anywhere to apply GSFLOW, and the implementation of topologically based domain discretizations, which may afford computational advantages for certain applications. GSFLOW-GRASS complements a software package recently released by the USGS that uses ArcGIS tools to set up regular grid domains for GSFLOW (Gardner et al., in review) . Providing different model implementation options affords greater flexibility depending on a user's software preferences and data formats. The coincident release of these two modeling support packages opens opportunities for rigorous testing of different model discretization and implementation strategies that can benefit different research and resource management applications.~~

We have demonstrated GSFLOW-GRASS ~~with~~ 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 ~~intensely farmed upper midwest~~ 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 ~~offers~~ is a flexible tool for investigating the role of groundwater-surface water interactions in modulating ~~dry season~~ dry-season discharge, controlling runoff in erosion-prone landscapes, and imposing possible ~~water quality~~ water-quality threats in agricultural and recreational watersheds.

We designed 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 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 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 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 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 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.

References

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

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

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

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.

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.

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

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

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

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

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

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

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

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

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

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

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

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, in review.

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

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

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

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

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

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

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

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

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

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

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

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

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

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.

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.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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.

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.

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.

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.

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

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

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

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

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.

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

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

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

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

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

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.