Gsflow-Arcpy - Python Toolkit for GSFLOW Model Development

Introduction

This manuscript describes a toolkit of Arc-python codes, known as Gsflow-Arcpy, that can be used to develop input data required for the U.S. Geological Survey (USGS) coupled Groundwater and Surface-Water Flow (GSFLOW) numerical, hydrologic simulation code, which is an integration of the USGS Precipitation-Runoff Modeling System (PRMS) and Modular Groundwater-Flow (MODFLOW) simulation codes. This toolkit does not produce all of the input data required for GSFLOW; however, it produces all of the data requirements for constant-area square HRU PRMS models, along with several input datasets required for MODFLOW simulations. Although PRMS has several modules for distributing precipitation, temperature, and solar radiation, and for simulating evapotranspiration, runoff, and streamflow, Gsflow-Arcpy generates input datasets for use of one set of these modules, including precip_1sta, precip_laps, temp_1sta, temp_laps, ddsolrad_hru_prms, potet_jh, srunoff_smidx, and strmflow, respectively. Datasets required for alternative PRMS modules are not supported in the first release of the toolkit.

Gsflow-Arcpy supports development of PRMS models, and the MODFLOW model top for constant area square MODFLOW cells for the Discretization (DIS) Package, the model exterior boundary array for the Basic (BAS) Package, and the stream network input for the Streamflow-Routing (SFR2) Package. Assuming the MODFLOW fishnet grid matches that of the PRMS HRU grid, these datasets can be used to create other MODFLOW input data that are derivative of these basic datasets. The toolkit works in tandem with the Cascade Routing Tool (CRT) to develop the input required for the PRMS Cascade Routing Module. The toolkit also can be used for applications that are not using PRMS by using a subset of the python scripts, such as for MODFLOW models that include the SFR2 Package applied to basins with complex topography. The toolkit consists of a series of python scripts that, when run in succession with required input datasets, will output Parameter Files ready for running PRMS, and a parameter shapefile that can be used to visualize results throughout the process and extract the required MODFLOW datasets. The PRMS Data File and Control File are not created by the toolkit and must be created separately, but templates are provided with the toolkit. Gsflow-Arcpy could be applied to other modeling applications, with some alterations to the model input files, but that is outside the scope of this tutorial.

The toolkit was developed to provide automation and reproducibility for creating GSFLOW models within the CONUS. However, some aspects of the development process, especially for developing the input required for the DIS Package, SFR2 Package, and Cascade Routing Module are difficult to fully automate in some cases. It may be necessary for the user to intervene within the automated procedure, and to manually modify the elevation of grid cells created by the toolkit within the ArcGIS environment. Additionally, it is recommended that the flow lines for the stream network provided by the NHD dataset be used as a guide for creating the SFR2 Package stream network. NHD stream network is not used directly to create SFR2 input due to inconsistencies between the resampled DEM used for creating model top and the NHD stream network. Where stream lines generated by the toolkit deviate significantly from the NHD streams, the user may need to modify the altitude of the model top data in order to create drainage directions that follow the NHD streams. Thus, especially in complex terrain, developing the stream network and cascades is an iterative process that includes: 1) filling closed depressions in the model top array using the CRT program; 2) generating the stream network and comparing to the NHD stream network; and 3) making local adjustments to the elevation of individual grid cells, and re-generating the stream network. This process may need to be repeated 2-3 times to generate a consistent model top, SFR2 stream network, and set of cascade parameters.

Gsflow-Arcpy consists of a set of Arcpy scripts that use tools supplied by ArcMap and associated add-in features, including ArcHydro. However, the toolkit scripts should be run outside of and while the ArcMap software is not running. If ArcMap is running in the background, the scripts may not run correctly, as they may not be able to acquire a lock on certain files. Thus, it is recommended that the toolkit scripts be run to generate output datasets, followed by loading the parameter shapefile into ArcMap. Datasets contained within the parameter shapefile can be displayed within ArcMap to evaluate progress and accuracy, as well as to make small modifications to HRU elevations to improve the generated stream network. The shapefile will need to be closed before running the following script, to regenerate the stream network, for example.

Previous work

Watershed modeling encompasses a vast number of processes and specifications, variable in time and space, and development of hydrologic models can be arduous. Despite the abilities of a process-based model, some may resist the technique due to practical developmental matters (Fatachi et al., 2016). Both research and management can benefit from hydrologic models, but their application is a complex process. A wider acceptance and distribution of process-based approaches will require improved model visualization tools and a streamlined approach for model setup, execution, and analysis (Fatachi et al, 2016). Integrated Hydrologic Models (IHMs) can provide important information about water resources and are often used as decision support tools for resource management (Laniak et al., 2013). Construction of IHM model inputs is not well formalized or automated, making reproducibility very difficult. A wider acceptance and distribution of process-based approaches requires improved model visualization tools and streamlined approaches for model setup, execution, and analysis (Fatachi et al, 2016). For example, various techniques have been developed to create drainage networks, but none have the benefit of a fully automated and topologically consistent approach (Turcotte et al., 2001).
The progression of IHMs has spurred research and development for improving data processing and graphical user interfaces (Band, 1986; Gruber and Peckham, 2009; Metz et al., 2011; Wilson, 2012; Tian et al., 2016).

The progression of watershed models has inspired research and development for improving data management and utilization with graphical user interfaces and geographical information systems (GIS) (Daniel et al, 2010). Advancements in GIS technology and remote sensing have provided the ability to highly parameterize systems of interest, but not without complex data processing and utilization. For example, PRMS requires upwards of 90 parameters. Creating input files requires complex GIS processing of raster and vector datasets from various sources. For example, developing a stream network employs several ArcHydro tools, while importation of soil and vegetation data calls for high level raster processing. Research has only recently begun to focus on GIS processing paired with model input construction. The GIS Weasel, described by Viger and Leavesley in 2007, provided the basic steps for developing input parameters for a watershed model, but required extensive data management and lengthy GIS processing. Alongside the GIS Weasel, PRMS had commonly been run through the USGS modular modeling system (MMS) that provided a platform upon which certain modules could be coupled to simulate a variety of processes (Leavesley et al, 2005). The GIS Weasel explains the tools to delineate, characterize, and parameterize features for distributed modeling. The MMS (no longer in existence) applied a model builder GUI that integrated process modules stored in a library. These primitive tools were valuable for combining the multiple components that contribute to the dynamics of a watershed, but required a high level of process knowledge and serious time investment on the front end of a modeling project. Together these platforms required the user to weave through a seemingly endless series of steps, tools, and phases.

Here we present a series of python scripts that provide a formalized technique for the model development of the IHM GSFLOW.. This python toolkit, called Gsflow-Arcpy, automates many of the necessary but laborious processes of parameterization, including stream network development, land coverage, and meteorological distribution over the model. Stream network development based on digital elevation models (DEMs) are needed in hydrology to determine the paths of water, sediment, and contamination movement (Tarboton, 1997). Similar techniques to those developed in older models such as TOPMODEL (Beven and Kirkby, 1979) and HYDROTEL (Fortin et al., 1995, 2001) that utilize a direction matrix and a compute topographic index are applied automatically in this toolkit through the use of ArcHydro tools (Maidment, 2002). The Cascade Routing Tool (CRT Henson et al., 2013) is utilized by Gsflow-Arcpy to provide cascade parameters for the streamflow routing packages (SFR, SFR2) used by GSFLOW.

Program and computing requirements

Mandatory requirements:

- ArcGIS 10.1 or newer (will include Python 2.7)
- Cascade Routing Tool (CRT) executable (version 1.3.0) http://water.usgs.gov/ogw/CRT/
- GSFLOW 1.1.6 or newer http://water.usgs.gov/ogw/gsflow/#downloads
- 4GB or more of RAM, depending on the size of the model.

Optional Requirements:

- USDA Soil Data Viewer and Microsoft Access http://www.nrcs.usda.gov/wps/portal/nrcs/detail/soils/survey/geo/?cid=nrcs142p2_053614
- A GitHub repository https://github.com/gsflow gives users the ability to download the latest versions of the python scripts and templates.

Geospatial and other ancillary data requirements

Input data required for GSFLOW models are derived from diverse geospatial data that are available for the continental United States and other places across the globe. These datasets are typically available through centralized data portals and can be downloaded freely to local computers for selected basins of interest, or by county. Gsflow-Arcpy does not access these geospatial datasets through web portals, rather they must be stored locally on the same computer as the toolkit. The toolkit develops the GSFLOW inputs by making calculations for each PRMS parameter and MODFLOW datasets according to methods described by Viger and Leavesley (2007), Markstrom et al. (2008), Markstrom et al. (2015), Harbaugh (2005), Niswonger et al. (2011), and Henson et al. (2013), Regan et al. (2017). Almost all of the data required by Gsflow-Arcpy can be downloaded for basins in the United States using the USDA Geospatial Data Gateway https://gdg.sc.egov.usda.gov/. Each particular dataset is explained individually below. In some cases, users may want to apply local datasets which may come from various sources. Datasets of this nature can be applied in the same manner as those obtained from USDA or USGS.

Elevation

Elevation data is typically in the form of 10m (1/3 arc-second) or 30m (1 arc-second) National Elevation Dataset (NED) rasters, which can be downloaded through Geospatial Data Gateway, or through the USGS National Map launch site (http://viewer.nationalmap.gov/launch/) under 'Download GIS Data'. The user will find several choices for elevation products, alongside a map to visualize and select tiles of these data. An FTP site, also under the 'GIS Data' section of the launch page, provides access to elevation data but without the clarity of the first option. If multiple digital elevation models (DEMs) are downloaded to cover the study area, they will need to be combined (via the mosaic tool in ArcGIS) into a single DEM for Gsflow-Arcpy processing. The elevation datasets will be resampled to the grid size resolution of the model based on user preferred statistics within the area of each model grid cell. LIDAR can be used if available. DEMs are generally available for the United States and across the globe, however, this availability and the resolution of these DEMs varies with location.

Study Area

The study area is a land polygon that defines the portion of an area of interest where hydrologic calculations will be made by GSFLOW. The boundary polygon will dictate the spatial projection of the model grid and the flow direction for each cell within the polygon. GSFLOW study areas may or may not correspond to the watershed boundaries, but all flow will be directed to the given model outlet points within the boundary. Where the study area is the same or nearly the same as the watershed boundaries, the study area boundary can be downloaded from the Geospatial Data Gateway. Watershed boundaries can be retrieved by selecting the "Hydrologic Units" option within the Geospatial Data Gateway website. Sub-basins contained within the study area can be developed from point shapefiles of downstream exit points, or pour points, in ArcGIS. If the study area is not available as existing watershed boundaries, ArcGIS tools can be applied to develop a polygon shapefile of the model domain using a DEM. Methods described herein or another approach can be used to develop the study area boundaries for use with the toolkit. Any land polygon dataset that defines the study area boundary can be used with the toolkit.

Vegetation

Spatially varying vegetation type and coverage must be defined within the study area for GSFLOW models. These data are provided by the LANDFIRE dataset, which can be downloaded as tiles or for the CONUS (http://www.landfire.gov/vegetation.php). Remap files are used to categorize the diverse sets of vegetation defined in the LANDFIRE US_120EVT and US_120EVC vegetation coverage datasets to a subset of vegetation types that are supported by PRMS (Viger and Leavesley, 2007). The US_120EVT and US_120EVC datasets can be downloaded for a study area as a single tile or as multiple tiles that must be aggregated to a single image before processing.

Soil

PRMS requires several parameters that are calculated from soil properties. Important soil properties that are used for parameterizing PRMS include the available water capacity (AWC), percent sand, silt, and clay, and saturated hydraulic conductivity (Ksat). These soil properties are provided for the CONUS by the STATSGO or SSURGO datasets, and can be downloaded through the USDA Geospatial Data Gateway. Typically, SSURGO data is used first, and if there are gaps in these data, STATSGO can be used to fill the gaps. Shapefiles of these soil properties can be extracted using the NRCS Soil Data Viewer tool and then converted to raster formats at the same resolution as the DEM for the study area. The Soil Data Viewer can be downloaded here, and installed as an add-in to ArcMap:
http://www.nrcs.usda.gov/wps/portal/nrcs/detail/soils/survey/geo/?cid=nrcs142p2_053614

Remap files

Text files with lists of the LANDFIRE vegetation codes and corresponding PRMS codes are necessary to re-assign LANDFIRE codes to the vegetation codes supported by PRMS. PRMS datasets derived from vegetation codes include coverage type, coverage density (summer and winter), snow interception, rain interception (summer and winter), and root depth. Example remap files are provided with the toolkit, and may need to be modified to include new or missing LANDFIRE vegetation types. Remap files will vary slightly depending on the version of ArcGIS being used, for example, Arc 10.1 allows the remap files to have strings after the values, but Arc 10.2 does not accept the strings. More notes on remap files and examples can be found in the appendix to this report.

Geology

To realistically distribute the saturated hydraulic conductivity (Ksat) of the soil, surficial geology maps can be applied. State maps are available for download here:
https://mrdata.usgs.gov/geology/state/
The rock type can be used to assign a multiplier to the Ksat in the equations concerning soil moisture, specifically regarding gravity drainage.

Impervious Cover

Impervious cover represents locations in the study area where the land surface is impervious to water, including, for example, paved areas or impervious bedrock outcrops. Impervious cover is used in PRMS to calculate runoff and infiltration. The impervious coverage data can be downloaded from the National Land Cover Database (NLCD http://www.mrlc.gov/nlcd11_data.php) for 2011. This dataset is intersected with the HRU grid to calculate the percent of impervious area per HRU.

Climate

The python toolkit produces the required PRMS parameters for use by the precip_1sta or precip_laps and temp_1sta or temp_laps modules. These parameters are used to distribute precipitation, and minimum and maximum temperature to all HRUs in the study area using daily climate data provided within the PRMS Data File. PRMS calculates daily precipitation at each HRU using daily values measured at one or more climate stations set in the PRMS Data File and average monthly precipitation adjustment factors (PRMS parameters rain_adj and snow_adj). The python toolkit calculates precipitation adjustment factors using the Precipitation Runoff on Independent Slopes Model (PRISM) data (Daly and others, 1994). Typically, 30-year monthly average precipitation at an 800m resolution is used, but longer periods of data from the PRISM group are available for a cost. Temperature adjustment factors, based on HRU aspect, (PRMS parameters tmin_adj and tmax_adj) used to calculate daily maximum and minimum temperature at each HRU are calculated by the Gsflow-Arcpy. PRISM data can be downloaded from the PRISM climate portal (http://www.prism.oregonstate.edu/normals/). Observed climate station data can be downloaded from several sources, depending on the study area. Typical station data for the Western U.S. are NRCS SNOTEL stations (http://www.wcc.nrcs.usda.gov/snow/snow_map.html) and WRCC NOAA coop stations (http://www.wrcc.dri.edu/climatedata/climsum/). Daily climate data across the CONUS can be found via the NOAA daily summaries map (https://gis.ncdc.noaa.gov/maps/ncei/summaries/daily). Local stations can be found to contain good data as well. If two climate stations at different elevations are available, precip_laps and temp_laps modules may be used, and PRMS will calculate daily lapse rates based on data at the two stations. If only a single station is used, monthly temperature lapse rates must be provided by the user, while precipitation distribution will be based off the relationship between station data and PRISM data.

National Hydrography Dataset (NHD)

A stream network provided by the NHD should be used as a guide when developing a stream network with Gsflow-Arcpy. The python toolkit uses the ArcHydro flow accumulation and flow direction tools to develop a stream network that is consistent with the model top dataset. For this approach, the user specifies minimum flow accumulation and stream length thresholds to create a stream network at a resolution appropriate for the modeling study. The flow direction is determined using the DEM resampled to the GSFLOW model grid. The DEM must have continuous downward sloping paths along the streams, such that every DEM cell in the study area is adjacent to a cell with lower altitude in 1 of the 8 adjacent cells, referred to as d-8 filling,. The Cascade Routing Tool (CRT) provides the capability to d-8 fill the DEM. Additional to the NHD stream network, aerial imaging of the study area is also helpful for verifying the location of stream lines in flat areas, or for canals that do not follow the natural drainage defined by topography. Aerial images are available through ArcGIS or can be downloaded through the Geospatial Data Gateway.

Configuration File (.ini) and Field List

The Configuration File for the python toolkit is a text file (.ini) that provides site specific parameters and folder specific information that is read by each python script. The location of this file is considered the working directory, which will house the folders and files created by Gsflow-Arcpy. The Configuration File holds input and output filenames and locations, spatial projections, stream network development thresholds, mean monthly precipitation data, monthly lapse rates, CRT settings, and many other project specific settings related to parameter development and ArcMap operations. Several flags can be turned on or off in the Configuration File, depending on preferences of the user and project specifics. Folder locations and names of input and output rasters, shapefiles, and text files are designated throughout the Configuration File, allowing the python scripts to call and read or operate on these files. The purpose of the Configuration File is to eliminate changing the python scripts, requiring only changes to be made in the Configuration File for each new project. More detail on the Configuration File is explained throughout the tutorial section, and a template Configuration File is provided with the toolkit.

Another text file, field_list.ini, is located in the python script folder. This file contains a list of the parameter names (or fields) that will make up the attribute table of the parameter shapefile. Some of these field names differ from PRMS parameter names, but are typically very close to the actual parameter names. This list does not need to be changed, but must be present, and is provided with the toolkit. If the user wants to change the name of a field, they may, but it is not necessary, and would generally require changing the name in certain scripts as well. A table is provided with the toolkit that holds explanations of each field name in the parameter shapefile.

Python Scripts

The python scripts make up the bulk of the Gsflow-Arcpy toolkit. The scripts can be stored in a folder separate from the working directory. Along with the scripts, a template Configuration File, field list file, and two CSV files that hold dimension and parameter settings are provided. For each new project, it is recommended that the user copies the Configuration File and the CSV files to a separate directory, where the files will be edited specifically to each project. The Configuration File, stored in the working directory, will guide the scripts to the correct folder locations, and the CSV files will provide specifics for the PRMS Parameter Files. More detail is provided in the tutorial section about both the Configuration File and the parameter and dimension CSV files. As the scripts are run, folders and files will appear in the working directory. The parameter shapefile holds the model attributes and can be examined in ArcMap after each script to check progress. The following section gives a brief description of each python script. More detailed explanations and functionalities of the scripts are provided in the appendix.

fishnet_generator.py

The fishnet generator uses the fishnet tool in Arc to build the model top dataset. This model top grid is used for defining HRU elevations, areas, stream network, soil properties, MODFLOW layer bottom elevations, etc. This script reads cell size and study area boundary settings from the Configuration File. The fishnet will adapt the spatial projection of the boundary shapefile. The fishnet script is needed if no fishnet shapefile exists, therefore it is recommended but not mandatory. In some cases, the user may want to use a pre-existing grid, or a section of an existing grid. In these cases, the script does not need to be run, and the pre-existing grid will determine the projection and cell size. To snap to a pre-existing grid, the user must input X and Y snap points in the configuration file.

hru_parameters.py

The hru_parameters.py script constructs the initial fields for the attribute table in the parameter shapefile, based on the field_list.ini file. The attribute table will display each of these field names as column headers. These columns start out blank, and fill with data as the scripts are run. This script also designates HRU_TYPE (0 = inactive, 1 = active, 2 = lake, 3 = swale) based on the study area boundary and lake and sink shapefile, if lakes or sinks are present. Swale is the PRMS term for sink, referring to a cell that does not have a cascade outlet. Endorheic basins must contain a designated swale or CRT will attempt to fill cells in order to route water out of the basin. The lake shapefile is typically extracted from an NHD dataset, or can be manually created. If swales exists, or known closed basins are being modeled, a shapefile must be provided with the sink points. The parameter shapefile attribute table can be edited by the user.

dem_parameters.py

This script populates the elevation fields based on the original DEM (DEM_MEAN, DEM_MIN, DEM_MAX, DEM_ADJ, DEM_FLOWAC, DEM_SUM, DEM_COUNT, and HRU_ELEV). HRU_ELEV is altitude in feet. DEM_ADJ values represent the surface elevation resampled to the model grid resolution, and are the elevation data that the stream network and cascade parameters are developed from. They are eventually written to a PRMS Parameter File as the hru_elev parameter. As these elevation based parameters are being developed, dem_parameters.py runs a flow accumulation function to weight the elevation of streams heavier than the surrounding elevations, in order to better represent stream altitudes at the model grid scale. If the original DEM is a smaller resolution than the model grid, these values get written to DEM_FLOWAC. The default setting in the Configuration File is to set these values as the original DEM_ADJ. The user can change this setting to give DEM_ADJ values from any of the DEM parameters, but the DEM_FLOWAC values are recommended, especially in complex terrain. This also provides an original model scale DEM column in the parameter shapefile that can be used as a fall back if mistakes are made while editing the DEM_ADJ values.

dem_2_streams.py

This script uses DEM_ADJ values to develop the gridded stream network. This is the first step in the iterative process of stream network development. The script operates Arc Hydro tools (flow accumulation and direction), based on thresholds set in the Configuration File, to build the initial stream network. The thresholds allow the user to obtain a desired level of detail in the stream network (figures 10 & 11). Once again, flow direction from all cells within the boundary shapefile will be forced to the given model points (outlet, sub-basin, or swale). Common issues with the stream network can include dead ends, misalignment relative to the NHD lines or aerial photos/basemaps, incorrectly located confluences, or unnecessary parallelization of streams (typically in flatter area). These issues can be fixed by either manually adjusting the elevation (DEM_ADJ), or by running crt_fill_parameters.py and using the CRT fill values to adjust DEM_ADJ. A combination of these methods can be used to route the streams correctly.

crt_fill_parameters.py

This script runs the Cascade Routing Tool (CRT) without streams and will identify undeclared swales and honor declared swales. The script will write CRT fill values to the fishnet (CRT_FILL), but won't modify the DEM_ADJ values unless the fill flag is TRUE in the Configuration File. This step can be left for the user to do manually, allowing for more user intervention. In the case that the user wants to use some of the CRT fill values, selected sections of the CRT_DEM column can be copied directly to the DEM_ADJ column. It is recommended for to users evaluate the swale areas and make manual adjustments, especially if the fill values are relatively large. After editing the DEM_ADJ values, the dem_2_streams.py script must be re-run to re-build streams based on the updated DEM_ADJ values, followed by re-running this script (crt_fill_parameters.py). A few iterations may be necessary to eliminate swales and route all streams correctly. Since the CRT does not fill cells that have a stream segment in them, running CRT without streams first ensures that there are no swales along the streams and that there is a continuous downward slope from cell to cell in the downstream direction.

stream_parameters.py

The stream parameter script runs the CRT with streams in order to generate the cascade routing parameters for PRMS. The script also generates the streamflow routing (SFR) input values for MODFLOW, including flow direction, reach (KRCH, IRCH, JRCH, IREACH), reach length (RCHLEN, MAXRCH), segment (ISEG, IRUNBOUND, OUTSEG, IUPSEG), and slope (STRM_SLOPE). These data are all added to the parameter shapefile and can be seen in the attribute table. If any edits to DEM_ADJ take place after this script has run, all three stream building scripts should be re-run, starting with dem_2_streams.py.

veg_parameters.py

The vegetation parameters script generates all PRMS parameters derived from the LANDFIRE vegetation dataset, including coverage type, winter and summer coverage density, winter and summer interception (snow and rain), and root depth. These parameters are written to the parameter shapefile. This script must be run before the soil scripts in order to establish root depths.

soil_raster_prep.py

The soil preparation script projects and clips the soil rasters to the model grid, and fills no data cells in the input rasters in order for soil_parameters.py to run smoothly. It is recommended that STATSGO data is used to cover any areas where SSURGO data is unavailable, but in the case that no data areas exist, this script will use neighboring soil types to assign values to the no data areas. This script must be run previous to soil_parapeters.py.

soil_parameters.py

The soil parameter script uses the prepared soil rasters to generate percentages of sand, silt, and clay, as well as available water content (AWC), saturated hydraulic conductivity (Ksat), and soil type. Soil depth will also be derived if a raster is provided, but is not required, as the greater of root depth and soil depth will be used for soil calculations. The script writes these values to the parameter shapefile and uses them to calculate PRMS soilzone parameters soil_moist_init, soil_moist_max, soil_rechr_init, soil_rechr_max, ssr2gw_rate, slowcoef_lin, slowcoef_sq.

prism_parameters.py

The PRISM parameter script determines a maximum and minimum temperature along with a precipitation value for each model grid cell for each month. Values are typically based on PRISM 30 year monthly normals (4km or 800m), but could be from another gridded climate dataset if available, and would have to be formatted to match that of the PRISM data. The script operates ArcMap's zonal statistics tool to resample the PRISM grid data to the model grid cells. This script must be run before the precipitation ratio script (ppt_ratio_parameters.py) is run.

ppt_ratio_parameters.py

The precipitation ratio script calculates values assigned to each grid cell for each month that acts as a multiplier based on the ratio between the PRISM gridded data and observed monthly mean precipitation values from a designated climate station (or multiple stations). These ratios (labeled as PPT_RT_01, PPT_RT_02, etc in the parameter shapefile) are the monthly rain and snow adjustment parameters (rain_adj, snow_adj) that PRMS applies to each HRU to account for elevation differences throughout the model domain.

impervious.py

The impervious script uses impervious cover data from the National Land Cover Database to derive the percentage of impervious surface within each HRU grid cell. The script writes the impervious percentage values to the IMPERV_PCT column in the parameter shapefile, which provides the values for the hru_percent_imperv parameter in PRMS.

prms_template_fill.py

The final script in the toolkit, template fill, writes the complete PRMS Parameter Files. Values written to the Parameter Files are directly read from the parameter shapefile that has been developed through the use of the Gsflow-Arcpy toolkit. The PRMS dimensions and parameters CSV files are also read by this script. The dimensions CSV file holds basic settings, several of which the user can adjust based on the project, such as number of climate stations (ntemp), for example. The parameter CSV holds the name of all parameters and designates which Parameter File they will be written to. Templates of these files are provided with the toolkit, and further explanation is provided in the tutorial section.

Notes on scripts

Scripts are generally run in the order that they are listed above; however, the scripts may be run in slightly different sequences. When the scripts are being used solely for generating a stream network for a MODFLOW model, for instance, the user may not need to prepare any files or run scripts related to vegetation or soil. Or, if the user is interested only in developing precipitation ratios, the PRISM scripts (prism_parameters.py and ppt_ratio_parameters.py) can be run before a stream network or land use parameters are developed. The fishnet_generator.py script must be run first for applications (assuming no pre-existing grid), followed by the hru_parameters.py and then dem_parameter.py scripts, as these define the model top HRU grid, parameter fields, and elevations.

Another python file, support_functions.py, exists within the toolkit. This script contains functions that are used by multiple scripts. This file must be present in the directory with the rest of the python files, but never needs to be explicitly run.

Tutorial Section

Example watershed – Russian River, Mendocino and Sonoma Counties, California, USA

This tutorial will guide the user through the process of applying the Gsflow-Arcpy toolkit to help develop a GSFLOW model of the Russian River basin, located in northern California, USA (figure 1). The user is encouraged to replicate the example model, or follow along using another area of interest. The Russian River basin was chosen to showcase the Gsflow-Arcpy capabilities within a large, dynamic watershed, but smaller examples are available with the toolkit as well. These small examples are tailored to specific model scenarios, such as a closed basin or a basin with a lake. These examples will run quickly and the user can compare results easily. The toolkit requires some amount of input file development in ArcGIS, which is briefly reviewed but not explained in any great detail, keeping the focus on the python tools. Plenty of outside resources exist for ArcMap operations.

In the case where only a stream network is desired, the user can develop the necessary fishnet parameters and build the stream network without running through the entire toolkit. These steps are explained in section A. Section B explains the remaining scripts that allow for complete development of PRMS Parameter Files, along with several MODFLOW input parameters. The tutorial describes the process of developing the model top grid (HRUs), grid cell altitudes, stream network, land use and coverage, soil characteristics, and climate data. When complete, the user will have a parameter shapefile containing the data necessary to create PRMS and MODFLOW input files that can be then applied as a GSFLOW model. Gsflow-Arcpy saves the user an incredible amount of time developing these data, allowing for more time to be spent on more significant tasks, such as model calibration, application, and analysis.

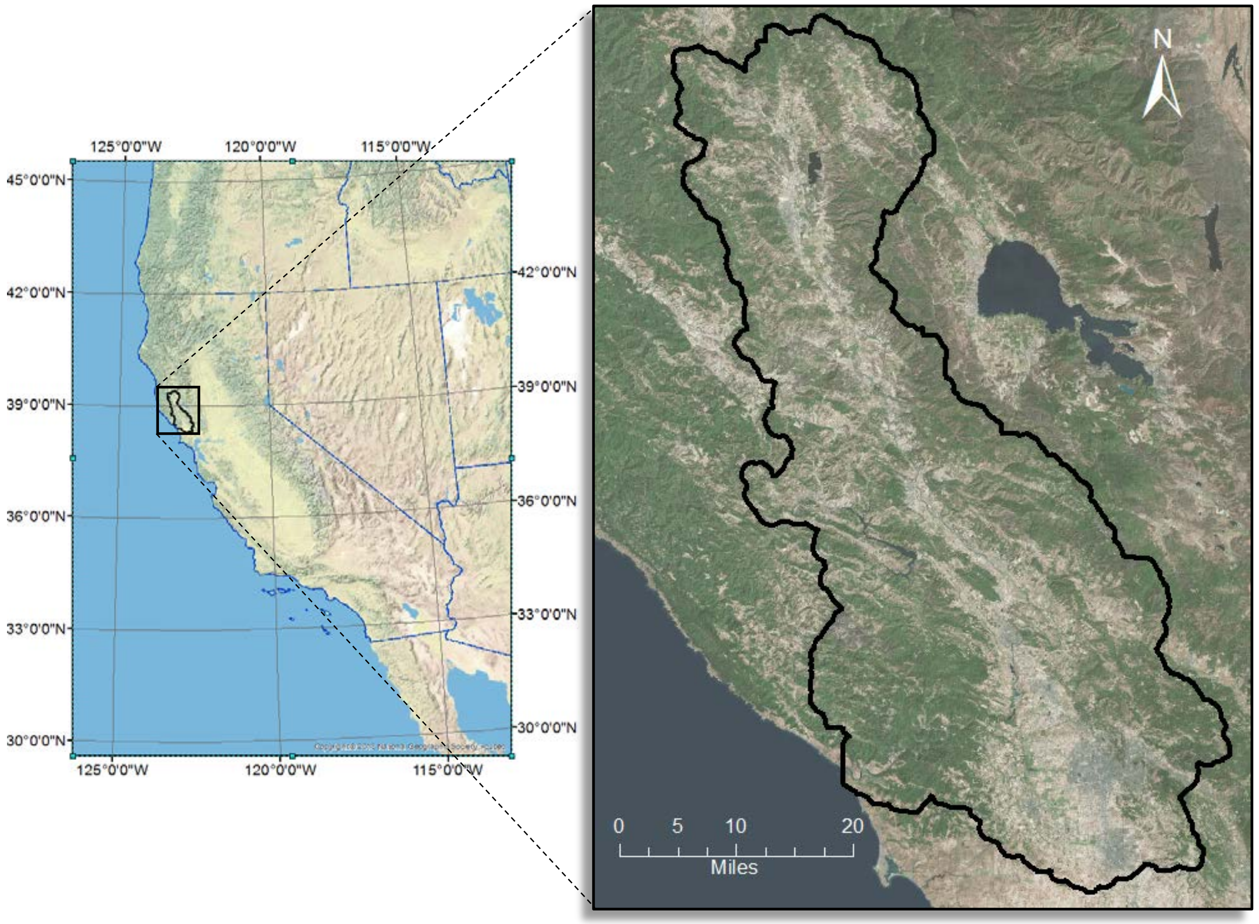


Figure 1: Russian River watershed, geographic location and hydrographic border

A. Develop fishnet grid, DEM parameters, and build stream network

I. Fishnet grid development using fishnet_generator.py

Step 1. Download a digital elevation model (DEM): In this tutorial, the National Elevation Dataset (NED) 10m (1/3 ac-second) is being used. Elevation data was downloaded in an IMG format through the USGS National Map Viewer launch site (https://viewer.nationalmap.gov/launch/) in 4 tiles that were aggregated in ArcMap using the mosaic tool to become one single raster image (figure 2). The four 1x1 degree IMG files that cover the Russian River basin are:

USGS NED n39w123 1/3 arc-second 2013
USGS NED n39w124 1/3 arc-second 2013
USGS NED n40w123 1/3 arc-second 2013
USGS NED n40w124 1/3 arc-second 2013

Download tiles to cover the area of interest, import into ArcGIS and use the mosaic tool to patch the tiles together, if necessary. This process generates a single DEM covering the study area. A study area boundary is not needed for this step, and it is recommended to download tiles to cover the entire area of interest and beyond to ensure that the entire study area is completely encompassed within the DEM. Save the DEM image file and set the location of the file to 'dem_orig_path' in the Configuration File in the 'DEM Parameters' section. The DEM file for this tutorial is named 'ned10m_nad83z10.img'.

Step 2. Download study area boundary: There are several ways to do this, depending on the study area. Some watersheds may need to be extracted from larger datasets, while other sources provide datasets for individual watersheds. The Russian River watershed boundaries were downloaded from the Geospatial Data Gateway (https://gdg.sc.egov.usda.gov/GDGOrder.aspx), which contains country wide datasets that can be ordered by county. Boundary datasets can be found under 'Hydrologic Units' in the Geospatial Data Gateway, after the county or counties are entered. The datasets named "8 digit watershed boundary" (NRCS version), for both counties within the Russian River watershed (Mendocino and Sonoma), contained boundaries for the study area. The datasets named "10 and 12 digit" contained sub-basin scale watersheds, and were not used for this tutorial, but would be acceptable for models focused on local, sub-basin hydrology. The watershed boundary polygon encompassing the Russian River basin was selected and extracted from the Mendocino County dataset in ArcMap (figure 2). The shapefile file path and file name were set in the Configuration File (figure 3).

Download watershed boundary file(s), and export the study area boundary to an individual shapefile in ArcMap. Enter the location of this file in the Configuration File under the 'Study Area' section (*study_area_path*). If no boundary is available through online resources, a watershed boundary can be created in ArcMap using the DEM and an outlet (pour) point. ArcMap applies the flow direction and accumulation tools in the Hydrology toolbox, under Spatial Analyst Tools. A watershed boundary gets created above the designated outlet point, which can be manually created, or can be a location of an actual streamgage. More information on this process can be found through ESRI help online. http://desktop.arcgis.com/en/arcmap/10.3/tools/spatial-analyst-toolbox/how-watershed-works.htm). Be aware that the spatial projection of the boundary shapefile will dictate the projection of the model fishnet and therefore all resulting rasters and data. The boundary will also control all flow directions, meaning that if the boundary is drawn outside the watershed, the elevations of those cells will be raised so that flow can be directed to the model outlet point.

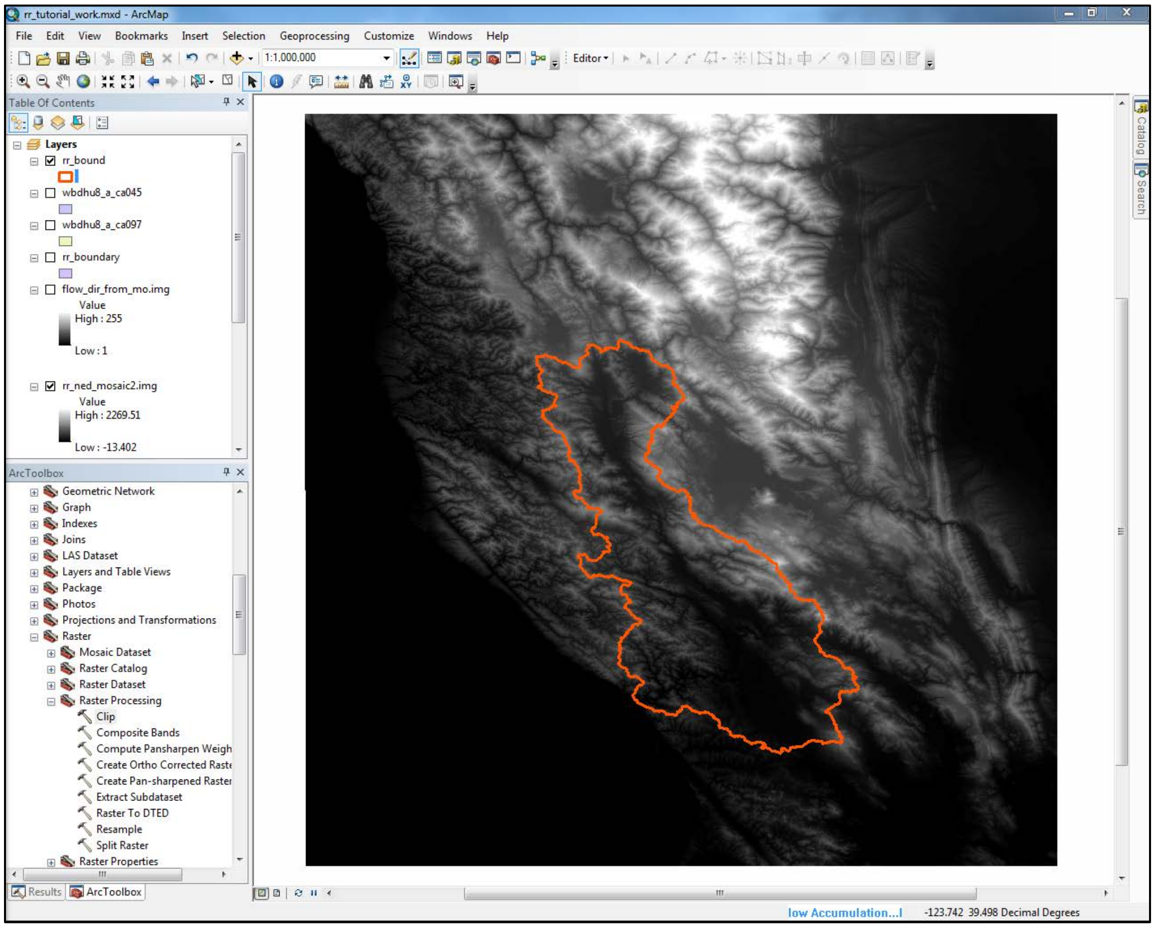
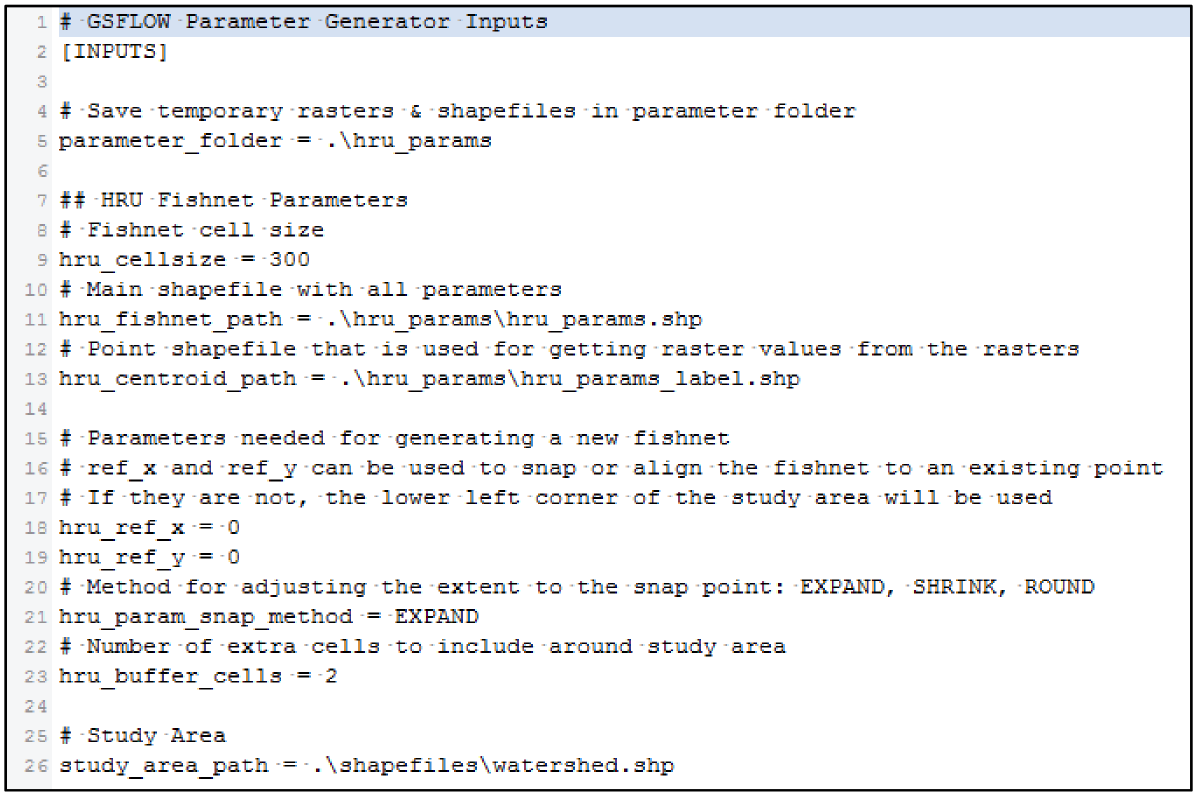


Figure 2: The combined DEM and Russian River watershed boundary (8-digit), visualized in ArcMap. These two datasets are the building blocks for Gsflow-Arcpy.

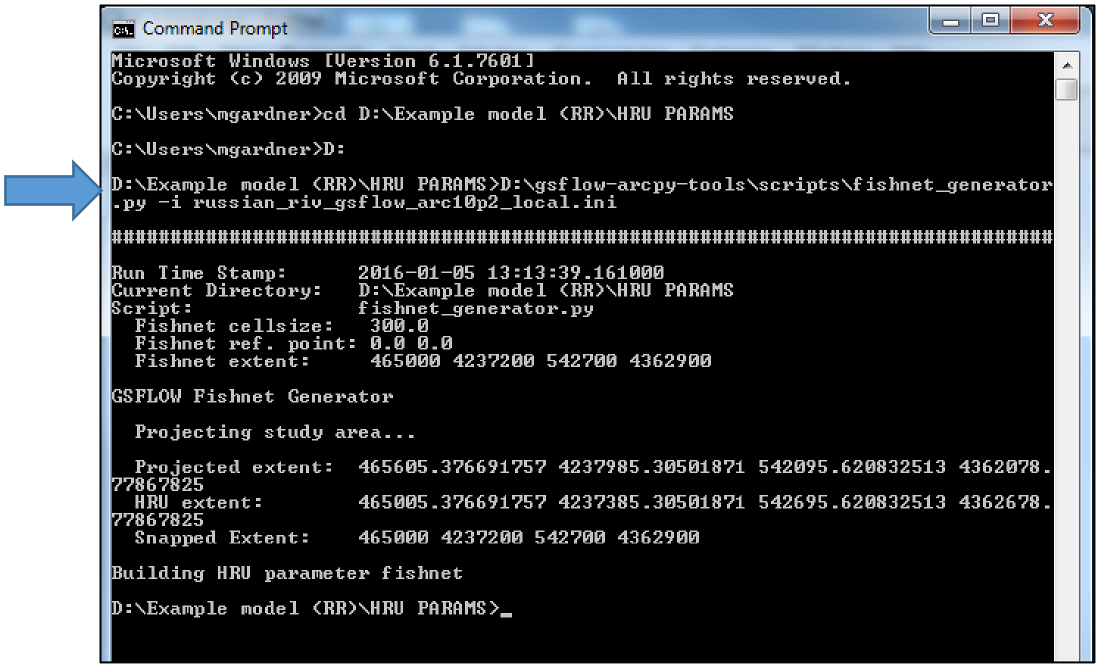
Step 3: Setup Configuration File: The Configuration File contains several settings that apply to the scripts, and also guides each script to specific files it needs to read. At the very least, the section related to the script being currently run must be filled out prior to running that script. For the fishnet section, set the desired grid cell and buffer cell size, and designate the folder path of the DEM and study area boundary. Buffer cells size designates how many layers of cells will be constructed outside of the model boundary. It is recommended to use at least 2 buffer cells. Smaller cell sizes cause longer runtimes for the scripts, so a balance must be found between model discretization and efficiency. This Russian River model was created with a 300 m x 300 m cell size. For the DEM to be re-sampled efficiently to the fishnet grid size, it is recommended that the cell size be set as a multiple of the DEM resolution. The boundary shapefile will provide a spatial projection and location for the fishnet. This will ensure spatial congruency for following operations. If a part of or entire pre-existing grid is to be used, the user may include a reference X-Y point in the Configuration File so the grid will snap to a specific location. The fishnet_generator.py script is not necessary in this case, and the spatial projection will be based off the grid provided. If no reference point is given, the lower left corner of the study area, plus buffer cells, will be used. Be sure to enter the folder and file names that will be given to the parameter shapefiles. The *hru_fishnet_path* will be the feature class shapefile that holds all model attributes, while the *hru_centroid_path* will be the point shapefile that holds centroid locations of each HRU for zonal statistics operations. An example section of the Configuration File and the settings used for the Russian River can be seen in figure 3. Several sections of the Configuration File will be shown throughout this tutorial, and an example template is provided with the toolkit.



```
# GSFLOW Parameter Generator Inputs
[INPUTS]

# Save temporary rasters & shapefiles in parameter folder
parameter_folder = .\hru_params

## HRU Fishnet Parameters
# Fishnet cell size
hru_cellsize = 300
# Main shapefile with all parameters
hru_fishnet_path = .\hru_params\hru_params.shp
# Point shapefile that is used for getting raster values from the rasters
hru_centroid_path = .\hru_params\hru_params_label.shp

# Parameters needed for generating a new fishnet
# ref_x and ref_y can be used to snap or align the fishnet to an existing point
# If they are not, the lower left corner of the study area will be used
hru_ref_x = 0
hru_ref_y = 0
# Method for adjusting the extent to the snap point: EXPAND, SHRINK, ROUND
hru_param_snap_method = EXPAND
# Number of extra cells to include around study area
hru_buffer_cells = 2

# Study Area
study_area_path = .\shapefiles\watershed.shp
```

Figure 3: A portion of the Configuration File that directs fishnet characteristics and provides the study area path.

To run any script in the python toolkit, the DOS command prompt is typically used. In the command prompt, the directory needs to be changed to the folder containing the Configuration File (the working directory). The folder containing the scripts must then be entered after the working directory, along with the name of the script being run, followed by a ‘-i’ and the Configuration File name (figure 4). A ‘-d’ typed after the Configuration File name will run the script in debug mode which provides more detail on the activity as the script runs.



```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\mgardner>cd D:\Example model (RR)\HRU PARAMS

C:\Users\mgardner>D:

D:\Example model (RR)\HRU PARAMS>D:\gsflow-arcpy-tools\scripts\fishnet_generator
.py -i russian_riv_gsflow_arc10p2_local.ini

################################################################################

Run Time Stamp:      2016-01-05 13:13:39.161000
Current Directory:   D:\Example model (RR)\HRU PARAMS
Script:              fishnet_generator.py
  Fishnet cellsize:    300.0
  Fishnet ref. point: 0.0 0.0
  Fishnet extent:      465000 4237200 542700 4362900

GSFLOW Fishnet Generator

  Projecting study area...

  Projected extent:  465605.376691757 4237985.30501871 542095.620832513 4362078.
77867825
  HRU extent:        465005.376691757 4237385.30501871 542695.620832513 4362678.
77867825
  Snapped Extent:    465000 4237200 542700 4362900

Building HRU parameter fishnet

D:\Example model (RR)\HRU PARAMS>_
```

Figure 4: Command prompt for running the python scripts. Note the folder path for the script is followed by a '-i' and then the Configuration Filename, as seen on the line next to the blue arrow. A '-d' can be added at the end of the command to run the script in debug mode in order to see more details on the script operation.

Step 4: Run fishnet_generator.py: One this script has run, the fishnet grid will appear in the folder designated in the Configuration File, and the shapefile can be opened in ArcMap (named 'hru_params.shp' in this tutorial). The initial shapefile will contain only three columns at this point: FID, SHAPE, and ORIG_FID. This fishnet shapefile is the model top, each grid cell representing a PRMS hydrologic response unit (HRU). The attribute table will build upon itself as each script is run, holding parameter values for each cell.

After running fishnet_generator.py, open the newly created shapefiles in ArcMap, along with the boundary shapefile, and observe the results of the script (figure 5). Check that buffer cells sufficiently provide extra space outside the watershed border, so that no active cell is the outermost cell of the fishnet. This will help when developing the stream network. Open the point shapefile (named 'hru_params_label.shp' for this tutorial) and check that each point represents the centroid of one cell. This file is used for several steps throughout the process regarding zonal statistic operations, such as re-projecting the DEM to the fishnet. Close the parameter shapefiles in Arc before running the next script. (Note – closing the shapefiles doesn't guarantee the next script will be able to acquire a lock. It is recommended to either close ArcMap completely, or open a new blank map in Arc where the shapefile can be re-opened once the next script is done running.)

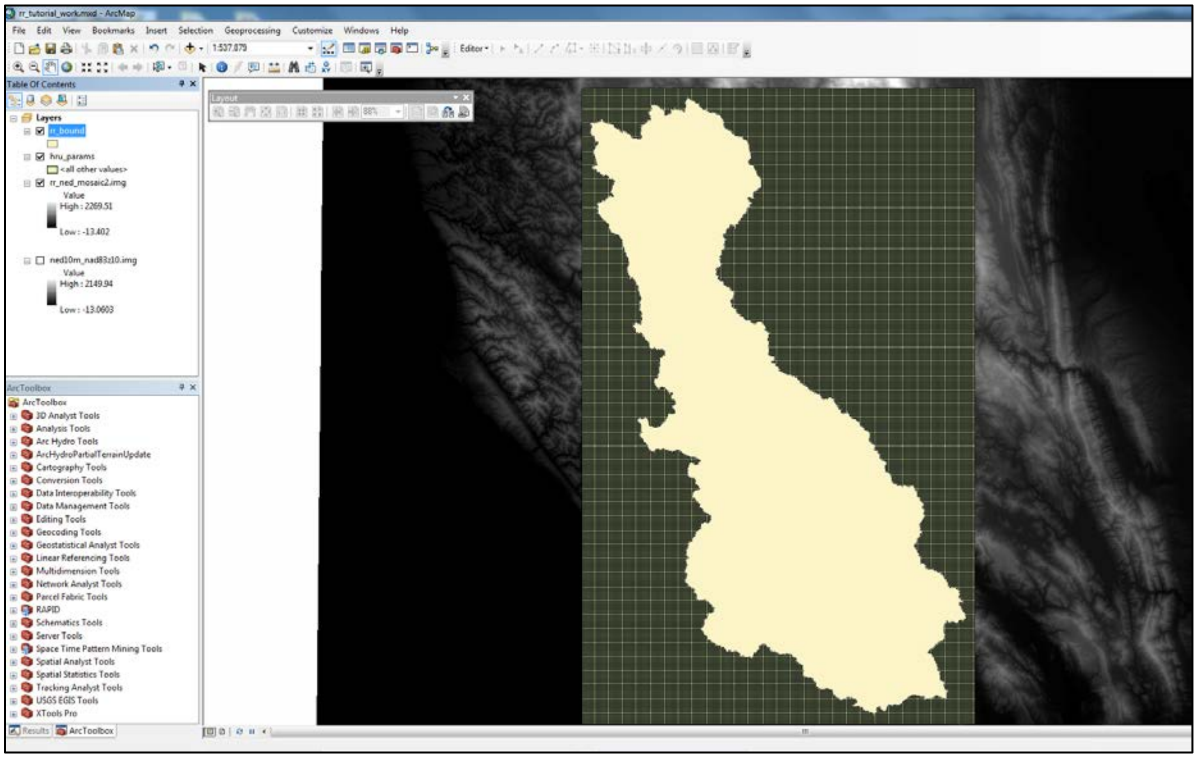
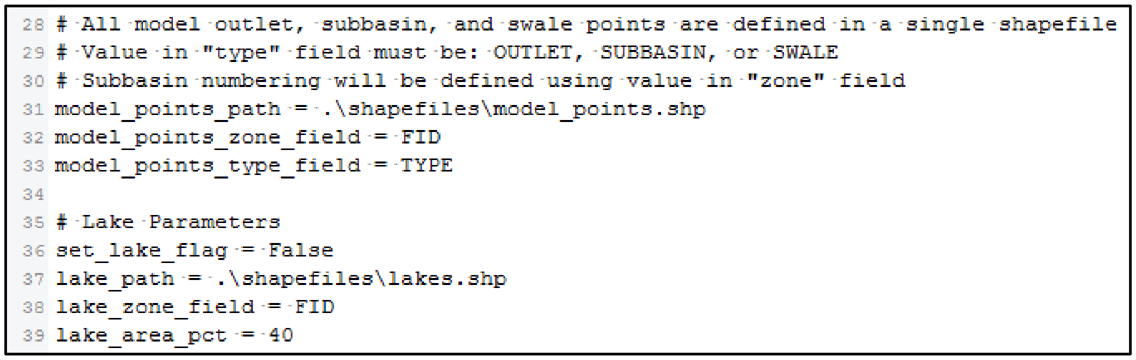


Figure 5: Fishnet shapefile open in ArcMap overlain by the Russian River watershed boundary polygon shapefile. Gsflow-Arcpy writes input data for PRMS and MODFLOW to the attribute table of the fishnet shapefile for each HRU grid cell.

II. Creation of HRU attribute table using hru_parameters.py

Step 1: Run hru_parameters.py: This script builds the attribute table for the HRU fishnet with field names for each parameter, as specified in the field_list.ini file provided with the toolkit. The script also designates HRU_TYPE: 0=inactive, 1=active, 2=lake, and 3 = swale. If lakes exist in the study area, a shapefile containing the lake polygons will be needed, with a filename and path designated in the Configuration File for *lake_path*. Water body data can be obtained from the National Hydrography Dataset (NHD) through the Geospatial Data Gateway. For the Russian River model, two major lakes were selected from the NHD and extracted, in ArcMap, to create a shapefile containing just the water bodies significant to the model. This file (rr_lakes.shp) is specified in the Configuration File as the *lake_path*. If no lakes are being modeled, set the *set_lake_flag* to false. To designate model outlet, sub-basin, and sink points, a point shapefile must be provided that contains a field with the point type designations OUTLET, SUBBASIN, or SWALE (figure 6).

All cells with an area >50% outside the study area boundary are declared inactive, and all cells with a greater area within the boundary are designated as active, or lake if more than half the cell area resides inside a lake boundary. This lake designation setting can be changed in the Configuration File (*lake_area_pct*), to account for a narrow section of lake.



```
# All model outlet, subbasin, and swale points are defined in a single shapefile
# Value in "type" field must be: OUTLET, SUBBASIN, or SWALE
# Subbasin numbering will be defined using value in "zone" field
model_points_path = .\shapefiles\model_points.shp
model_points_zone_field = FID
model_points_type_field = TYPE

# Lake Parameters
set_lake_flag = False
lake_path = .\shapefiles\lakes.shp
lake_zone_field = FID
lake_area_pct = 40
```

Figure 6: Configuration File clip regarding model points and lakes shapefiles. The field that contains the point type must be designated and sub-basin numbering will be based on the values in the field provided for model_points_zone_field.

After running hru_parameters.py, check the attribute table in the parameter shapefile to ensure the fields have been created and values have been generated for HRU_TYPE. Use the Categories display option to view HRU_TYPE and make sure the lakes and swales (if any) are in the correct location (figure 7).

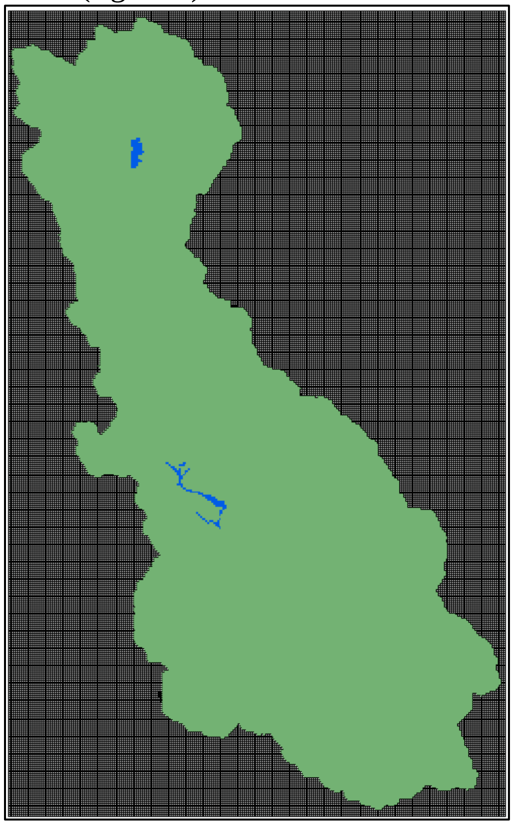
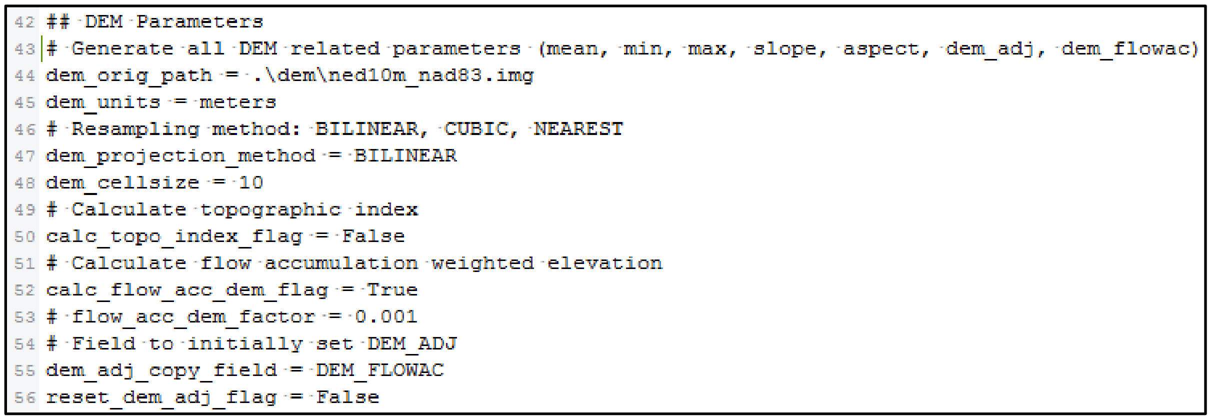


Figure 7: HRU fishnet symbolized with HRU_TYPE = 0 for inactive cells (blank), HRU_TYPE = 1 for active cells (green), and HRU_TYPE = 2 for lakes (blue). No sub-basins within the Russian River watershed are endorheic, therefore no swale points were provided. If so, these would appear in a different color and be HRU_TYPE = 3.

III. Write DEM-based parameters to attribute table using dem_parameters.py

Step 1: Designate DEM settings and file paths in Configuration File: The original DEM file path and associated settings need to be designated in the Configuration File, including *dem_orig_path* and *dem_cellsize* (figure 8).

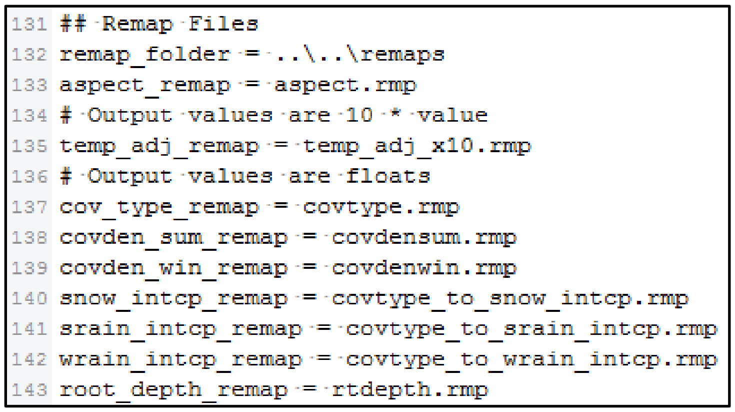


```
42 ## DEM Parameters
43 # Generate all DEM related parameters (mean, min, max, slope, aspect, dem_adj, dem_flowac)
44 dem_orig_path = .\dem\ned10m_nad83.img
45 dem_units = meters
46 # Resampling method: BILINEAR, CUBIC, NEAREST
47 dem_projection_method = BILINEAR
48 dem_cellsize = 10
49 # Calculate topographic index
50 calc_topo_index_flag = False
51 # Calculate flow accumulation weighted elevation
52 calc_flow_acc_dem_flag = True
53 # flow_acc_dem_factor = 0.001
54 # Field to initially set DEM_ADJ
55 dem_adj_copy_field = DEM_FLOWAC
56 reset_dem_adj_flag = False
```

Figure 8: Clip of the 'DEM parameters' section of the Configuration File.

As can be seen in figure 8, *dem_adj_copy_field* (line 55 in figure 8) is set to DEM_FLOWAC. As explained in the introduction, this setting designates which elevation based field will set the initial values for DEM_ADJ. Of all the elevation parameters, DEM_FLOWAC is the preferred option for most cases, especially in steep topography, as it provides better altitude values along streams during the resampling of the original scale DEM. However, the user may choose DEM_MEAN, for example, to provide initial values for DEM_ADJ.

Along with the elevation parameters, dem_parameters.py also calculates the aspect, slope, and relative temperature adjustment for each grid cell. The script reads aspect and temperature remap text files, the file paths of which are set under the 'Remap files' section of the Configuration File (figure 9). The temperature adjust remap file used for this example is "temp_adj_x10.rmp". The format of the remap file is for ArcMap 10.2 and 10.3. The remap values are multiplied by 10 in order to convert the floating point values to integers because the latest versions of ArcMap require integers when calculating the aspect and temperature adjustment. The script divides the values by 10 internally before generating the results. Remap templates are provided with the toolkit. The remap values should be checked by the user prior to running the script, and may be changed based on the study area.



```
## Remap Files
remap_folder = ..\..\remaps
aspect_remap = aspect.rmp
# Output values are 10 * value
temp_adj_remap = temp_adj_x10.rmp
# Output values are floats
cov_type_remap = covtype.rmp
covden_sum_remap = covdensum.rmp
covden_win_remap = covdenwin.rmp
snow_intcp_remap = covtype_to_snow_intcp.rmp
srain_intcp_remap = covtype_to_srain_intcp.rmp
wrain_intcp_remap = covtype_to_wrain_intcp.rmp
root_depth_remap = rtdepth.rmp
```

Figure 9: Remap settings in the Configuration File. The aspect and temperature adjust remap files are used by dem_parameters.py, while the rest are used in later scripts.

Step 2: Run dem_parameters.py: This script will re-sample the original DEM to the model grid cell size, and create all elevation, slope, and aspect related parameters. Once the script has finished running, the parameter shapefile can be opened in ArcMap to check and visualize the data that have been written to the attribute table (figure 10). Because this script will overwrite any existing DEM values, the user will see a message warning of this and will be prompted to press ENTER to continue running the script. This is a safeguard to avoid an unintentional overwrite of manual adjustments. If the user wishes to refresh the DEM and start anew, the script can easily be re-run, or the *dem_adj_copy_field* can be copied into the DEM_ADJ column.

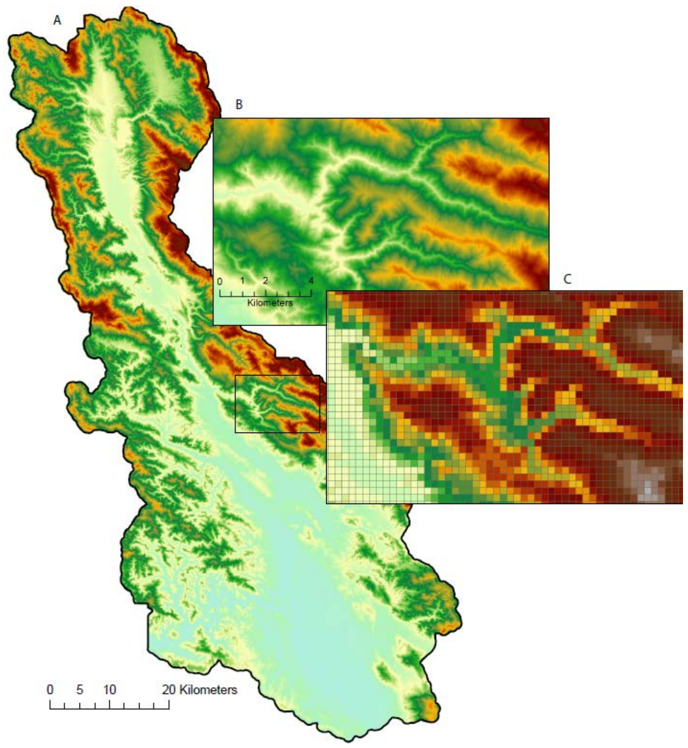


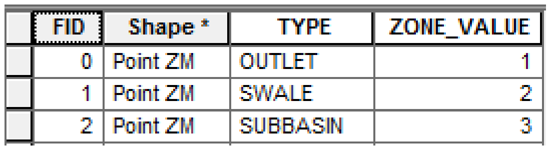
Figure 10: A) Original 10m DEM, clipped to the Russian River watershed, B) inset at the original DEM scale, and C) inset of the 300m DEM re-sampled for the model grid.

IV. Development of the stream network and cascade numbering using dem_2_streams.py, crt_fill_parameters.py, and stream_parameters.py

Development of the stream network is typically an iterative process. Aerial imagery and the NHD stream lines can be used to evaluate the accuracy of the stream network generated by Gsflow-Arcpy. The NHD is not used directly because it is not consistent with the DEM_ADJ data, and this causes unstable solutions for the SFR2 Package in MODFLOW. Filling the DEM using 8-way fill in ArcMap and the CRT modifies the DEM_ADJ dataset to provide 1) continuous slopes in the direction of streamflow for all grid cells that contain stream reaches, and 2) continuous drainage pathways for all HRUs within the study area domain, such that all cascade termination points end at a stream, lake, or specified swale. The change in elevation applied to each cell during the fill process should be evaluated by examining the fill rasters created in the 'flow_rasters' folder in the working directory. These are created by dem_2_streams.py.

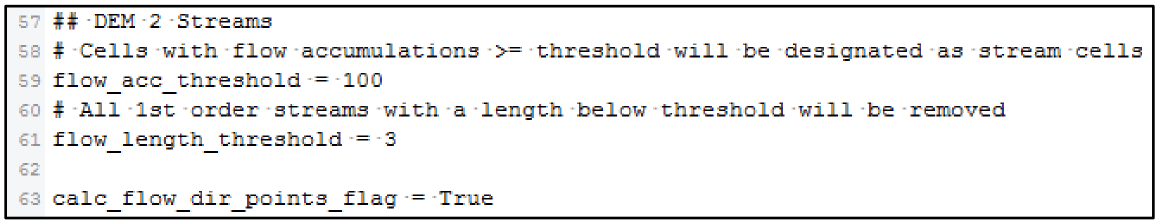
Solely applying 8-way fill does not always result in a satisfactory DEM_ADJ dataset for creating the stream network. In some cases, the altitude of cells in DEM_ADJ can be better modified by hand. Adjustments are made by hand when the stream network is not continuous or if stream lines do not follow closely enough the NHD stream lines or aerial imagery. After changes (fills) to DEM_ADJ are adopted from CRT results or made by hand, CRT should be run again to verify that no additional fill is required and to generate the cascades. If the fill process does not work well or the user is unhappy with the stream network, DEM_ADJ can be reset to the *dem_adj_copy_field* (DEM_FLOWAC in this tutorial), which holds the original DEM_ADJ values. The user would then apply different adjustments. Details on this process are described in the following steps.

Step 1: Run dem_2_streams.py: This script develops the initial stream network, based on DEM_ADJ. It also builds PRMS sub-basins if any model points are designated as SUBBASIN (figure 11). Sub-basins are defined using outlet points, or "pour points" as they are referred to in the ArcMap literature. These points can be actual streamgage locations or manually inserted points to designate the bottom (outlet) of a sub-basin. The script uses the model points shapefile and the watershed tool in ArcMap to delineate sub-basins. The script creates a stream network within the constraints of the flow length and flow accumulation thresholds set by the user in the Configuration File, in the DEM 2 Streams section (figure 12). The flow accumulation and flow length thresholds can be adjusted to attain a stream network with the desired level of detail (figure 13).



FID	Shape *	TYPE	ZONE_VALUE
0	Point ZM	OUTLET	1
1	Point ZM	SWALE	2
2	Point ZM	SUBBASIN	3

Figure 11: Example of a model points shapefile attribute table, created by the user to designate the model outlet point, sub-basin points, and swales. The ZONE_VALUE determines the basin numbering.



```
57 ## DEM 2 Streams
58 # Cells with flow accumulations >= threshold will be designated as stream cells
59 flow_acc_threshold = 100
60 # All 1st order streams with a length below threshold will be removed
61 flow_length_threshold = 3
62
63 calc_flow_dir_points_flag = True
```

Figure 12: Configuration File settings for the stream network. The *flow_acc_threshold* (line 59) and *flow_length_threshold* (line 61) can be adjusted to construct the desired level of detail of the stream network. The *calc_flow_dir_points_flag* (line 63) will build flow direction arrows if set to True. This is a nice visualization tool for the user to observe flow direction results.

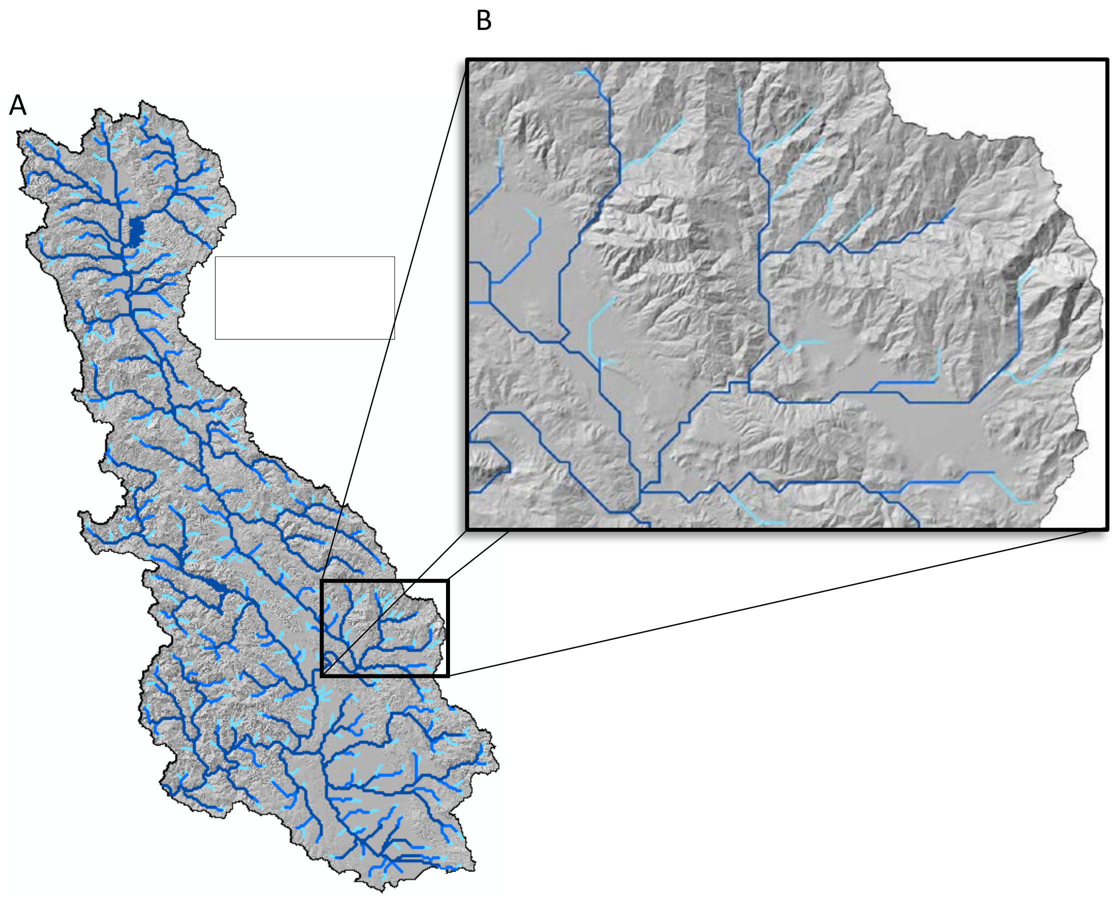


Figure 13: Stream networks generated with different flow accumulation and flow length threshold settings; A) Russian River watershed, and B) close up of a sub-basin showing different stream resolutions. The darkest blue lines (streams_100_10) show a network with a flow accumulation threshold of 100 and flow length threshold of 10. HRUs with flow accumulation and flow lengths smaller than the specified threshold will not contain streams. The dark and light blue lines show the levels of detail that can be obtained by decreasing the flow accumulation and stream length thresholds (streams_50_5 and

streams 30_3, respectively). These thresholds are site specific and depend on the resolution and purpose of the model.

In the case of an endorheic basin that does not have any outflow, a cell within the basin must be specified as a swale (HRU_TYPE = 3). This will have been assigned in hru_parameters.py. This will create a sub-basin with no outlet, where local streams will flow to the sink point. If a sink point is not designated, ArcMap and CRT fill algorithms will attempt to fill the local depression until an outflow exists.

Step 2: Observe initial stream network and modify DEM_ADJ: The dem_2_sreams.py script creates a stream shapefile (streams.shp) located in the 'flow_rasters' folder that gets created in the working directory. Import streams.shp into the ArcMap workspace, overlaying it on the HRU grid and basemap imagery, along with NHD streams for comparison. A hillshade map also can aid in evaluating the generated stream network. Evaluate the headwater drainages and tributaries to assess the effect of the flow accumulation and length thresholds on the level of detail in the stream network. Several threshold re-adjustments may be necessary, re-running dem_2_streams.py each time, to arrive at the desired stream detail. Zooming and panning throughout the model domain will reveal areas in the network that may require changes to the DEM_ADJ in order to clean up, smooth, or re-route sections of the network.

Image rasters that display the characteristics of the stream network are also created in the 'flow_rasters' folder with this script. Undeclared swales in the active model domain, named 'dem_sink.img', shows all cells that ArcMap fill algorithms found to need filling to create outflow. Other rasters created by this script include flow direction points, flow accumulation, stream links, stream order, stream count, sub-basins, and watersheds (per segment). The 'dem_fill.img' raster shows how much the DEM_ADJ values will have to be raised in each cell. These values are written to the DEM_SINK column in the parameter shapefile attribute table. In cases where fill values are relatively low (<5m), the user may choose to use computed fill values, therefore continuing on to run crt_fill_parameters.py and using the CRT_ELEV field to directly replace DEM_ADJ values in the swale locations. (The ArcMap fill and the CRT fill values should be very similar). However, in some cases, there will be areas within the model domain that can benefit from manual adjustments before continuing. Adjusting DEM_ADJ during this step should be done to fix issues with sinks or with the stream network before running the following stream scripts. An example of a manual edit of DEM_ADJ to alter the stream network can be seen in the following figures.

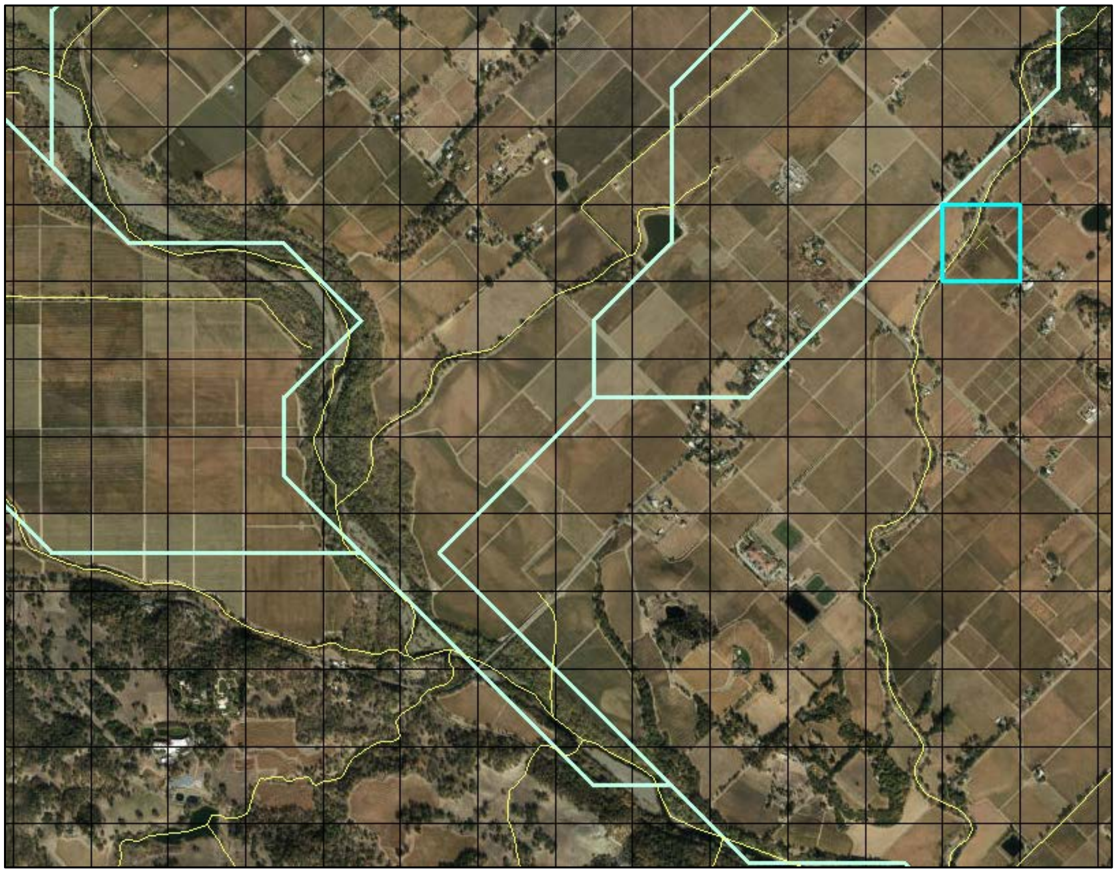


Figure 14: A clip of the initial stream network (streams.shp), shown in light blue lines, overlaying the HRU grid (thin black lines), NHD streams (thin yellow lines), and basemap imagery. The teal outline of a single grid cell shows where the elevation adjustment should take place to guide the stream back to its actual path.

It can be seen in figure 14 that the tributaries coming in from the top-right are not well represented by the initial stream shapefile. It is clear they diverge from the stream paths visible in the imagery and the NHD lines, and there is an unnecessary parallelization where the tributary meets the main stem. In this case, the user may choose to manually edit the DEM_ADJ values of some surrounding cell(s) in order to direct the streams more realistically to the main channel (running from top-left to bottom right in the figure). Users must be sure to avoid creating any swales while manually editing. Regarding the tributary approaching from the left side of the main channel, the user must decide if an adjustment here is necessary to follow the NHD stream path or not. This depends on each individual model and purpose.

Figure 15 shows the HRU grid cells labeled with the DEM_ADJ altitudes, and the cells with light green borders have been slightly adjusted (by hand) in order to re-route the stream to better follow the true streams seen in the imagery and NHD lines, and to eliminate the parallelization of steam segments seen in figure 14.

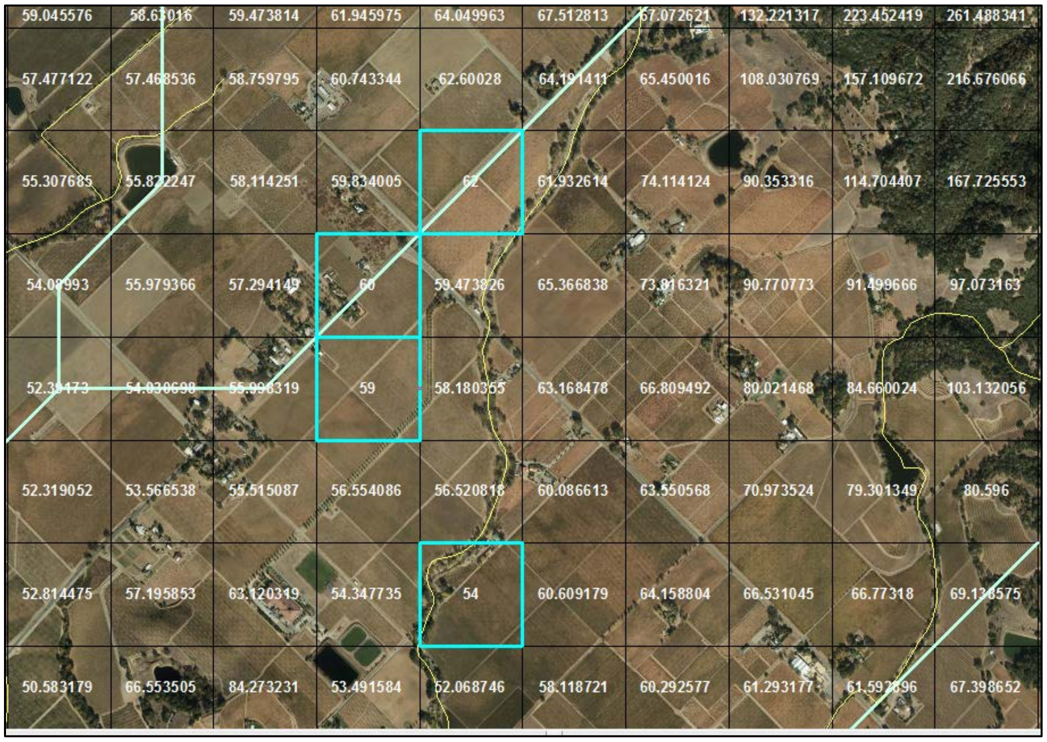


Figure 15: DEM_ADJ values labeled in each cell, and selected cells that have been slightly adjusted in order to re-route the stream to follow the actual stream path. The amount of manual editing of the stream network is left up to the user.

Once edits have been made and the user would like to see the results, save and close the project in ArcMap, and re-run dem_2_streams.py. Re-open the parameter and stream shapefiles and observe the changes in the stream network and any remaining sinks. The user may choose to edit several of the largest sink areas and perform some manual stream routing to clean the network up, and then rely on the CRT_ELEV values (created in the next step) to fill the rest of the swales throughout the model domain. Figure 16 shows the final result for this example section of stream network editing.

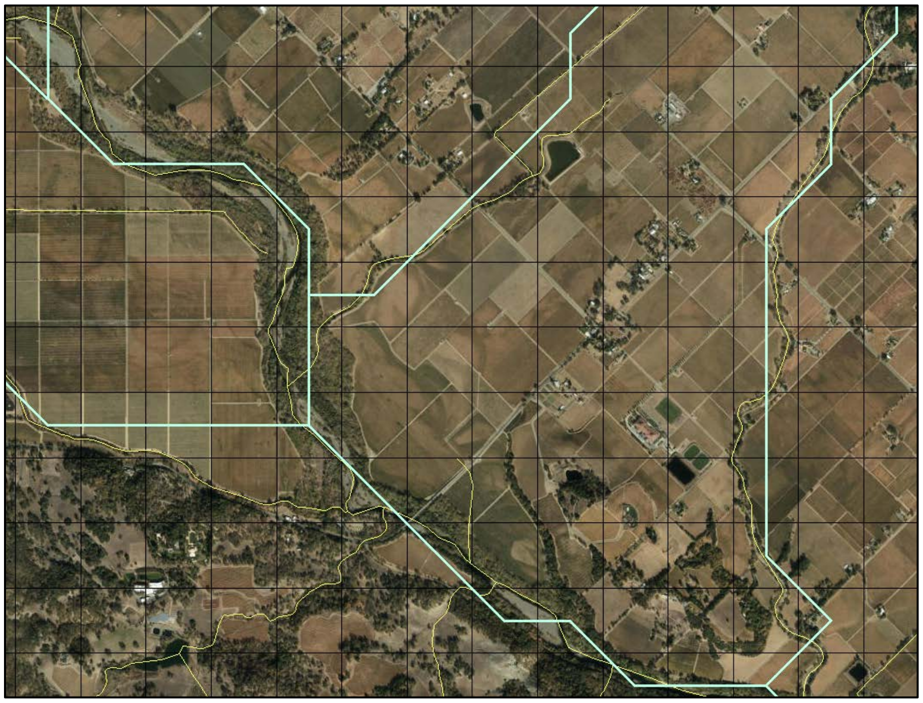
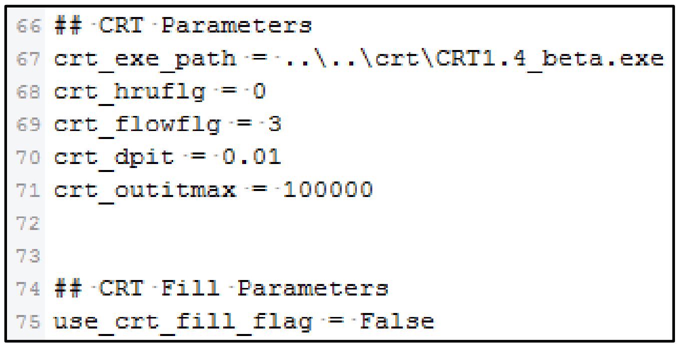


Figure 16: The resulting flow paths after manual adjustments. Note the cleaner, more realistic flow lines, resulting from a few small changes to select cells.

Step 3: Finalize stream network by running crt_fill_parameters.py: Once satisfied with the stream network and the magnitudes of the sinks, close the ArcMap project and run crt_fill_parameters.py. This step (and step 4) require the CRT executable to be located by the scripts, which must be downloaded by the user (link provided in introduction). Set a file path in the Configuration File under the "CRT parameters" section (figure 17). A folder will be created named 'fill_work' in the working directory, which will store the CRT input and output files, compete with swales and associated HRU cell ids. The parameter shapefile attribute table will contain fields named CRT_FILL and CRT_ELEV. CRT_FILL is the amount of elevation change needed to fill a swale, and CRT_ELEV is the elevation of the cell with that fill amount applied. Remember, in the case that a model domain contains actual swales that the user wants to model, HRU_TYPE can be designated as a swale if set to a value of 3 (Henson et al., 2013). This will keep the CRT fill from operating on that swale area. After observing the potential CRT fill amounts, the user can choose here to step back and perform manual edits to eliminate these swales by hand, or to simply copy the CRT_ELEV values to the DEM_ADJ column where they differ, and re-run both dem_2_streams.py and crt_fill_parameters.py. If the *use_crt_fill_flag* is True, the values from the CRT_ELEV will automatically be copied to DEM_ADJ. These steps will result in a stream network free of undeclared swales and ready for CRT to run with streams to build cascades (figure 18). Keep in mind that if any editing of the DEM_ADJ takes place, dem_2_stream.py must be run again, followed by crt_fill_parameters.py, so that stream_parameters.py will build cascades correctly.



```
## CRT Parameters
crt_exe_path = ..\..\crt\CRT1.4_beta.exe
crt_hruflg = 0
crt_flowflg = 3
crt_dpit = 0.01
crt_outitmax = 100000


## CRT Fill Parameters
use_crt_fill_flag = False
```

Figure 17: CRT settings in the Configuration File. The scripts run CRT internally, but need to locate the executable. Folders named ‘fill_work’ and ‘cascade_work’ contain CRT results that will be created in the working directory. The CRT parameters can be adjusted for someone familiar with CRT, but the template Configuration File will be preset to the recommended CRT settings. More information regarding the CRT settings can be found in Henson et. al., 2013.

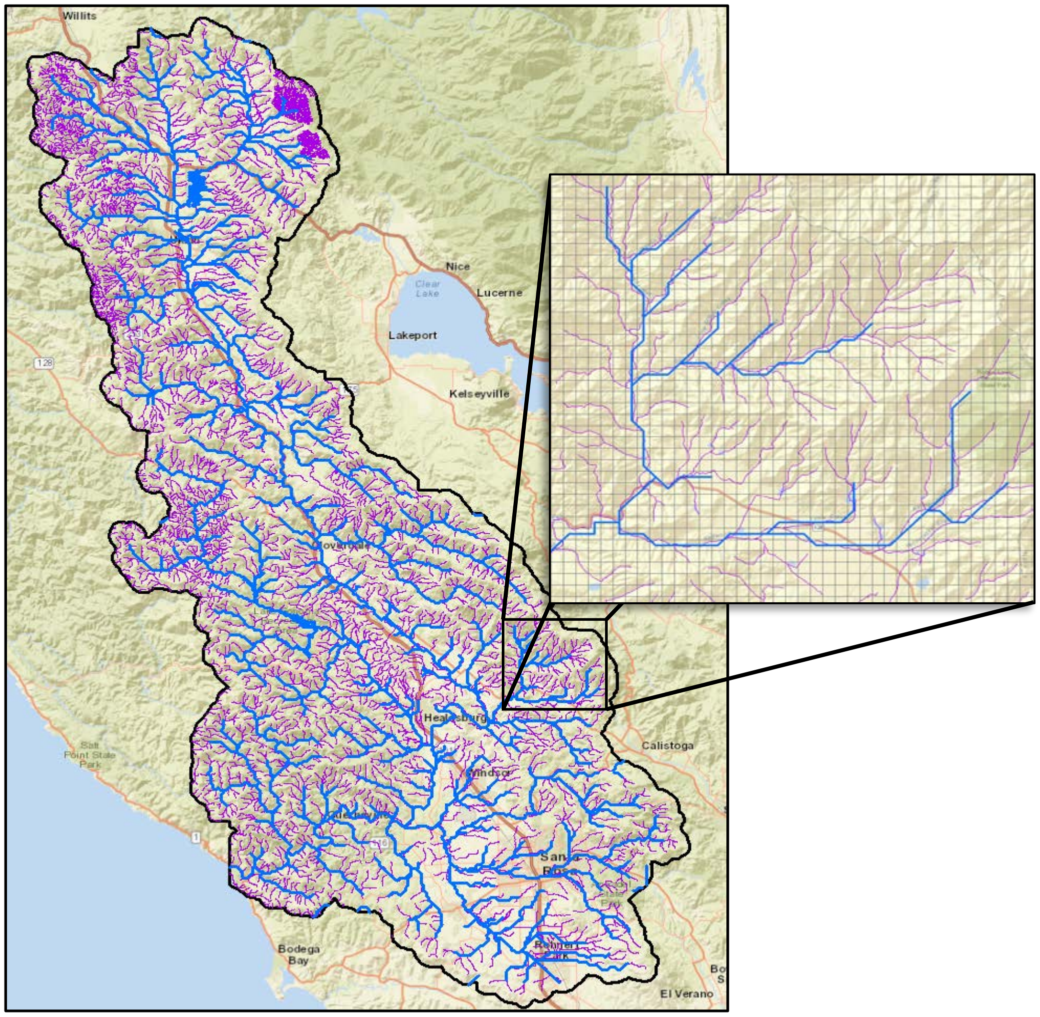


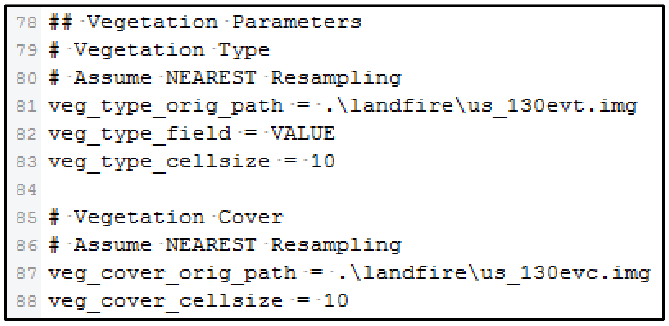
Figure 18: Stream network (blue lines) overlying NHD flowlines (purple), with the inset displaying grid cells (gray lines).

Step 4: Develop the PRMS and MODFLOW cascade parameters by running stream_parameters.py: The stream_parameter.py script will run CRT with streams and assign the final cascade numbering to each cell. This is the final script in the stream building process, and will result in the final cascade parameters for each HRU. It is not unusual iterate through these stream development scripts to eliminate all swales and develop the final stream network. For large watersheds, this process may take place in sections as the user moves through the watershed. For smaller models, only one or two iterations may be needed. In some cases, a simple model may not require any manual adjustments. In this case, the *use_crt_fill_flag* would be TRUE and the stream building steps would interate a single time, only repeating dem_2_streams.py and crt_fill_parameters.py once, after the initial sinks were filled. Results of stream_parameters.py can be observed by checking HRU_CASC.DAT, OUTFLOW_HRU.DAT, LAND_ELEV.DAT, XY.DAT and the 'outputstat.txt' file in the 'cascade_work' folder, and visualizing parameters in ArcMap (ISEG, IRUNBOUND, FLOW_DIR, KRCH, IRCH, JRCH, ISEG, REACH, RCHLEN, MAXREACH, OUTSEG, IUPSEG, SUBBASIN, SEGBASIN, STRM_TOP, STRM_SLOPE). The cascade parameters are used for input to the Streamflow Routing (SFR) package in MODFLOW.

B. Prepare remap files, soil, vegetation, and climate parameters, and create the PRMS Parameter File

I. Preparing remap files, vegetation data, and running veg_parameters.py

Step 1: Download vegetation type and cover: For the Russian River watershed, existing vegetation type (EVT) and cover (EVC) from the LANDFIRE data distribution site was used (http://landfire.cr.usgs.gov/viewer/). These datasets were stored to a local folder and the file paths were set in the Configuration File (figure 19). Attribute data dictionaries can also be downloaded (http://www.landfire.gov/NationalProductDescriptions21.php for vegetation type and http://www.landfire.gov/NationalProductDescriptions23.php for vegetation cover). The attribute data dictionaries help update the vegetation remap files for the study area.



```
## Vegetation Parameters
# Vegetation Type
# Assume NEAREST Resampling
veg_type_orig_path = .\landfire\us_130evt.img
veg_type_field = VALUE
veg_type_cellsize = 10

# Vegetation Cover
# Assume NEAREST Resampling
veg_cover_orig_path = .\landfire\us_130evc.img
veg_cover_cellsize = 10
```

Figure 19: Vegetation type and cover sections of the Configuration File. Cell size is the size at which the data will be re-sampled to the model grid, so a factor of the model grid cell size is recommended to ensure best alignment.

Step 2: Check/update remap files: It is recommended that the user scans through the remap files provided with the toolkit to ensure that all vegetation coverage types, densities, root depths, and interceptions are acceptable, referring to LANDFIRE vegetation data files if there are questionable vegetation types assigned in the study area. If the study area contains vegetation types that are not included in the remap files, they will need to be added to the remap list. Save these remap files to a folder and set the file paths in the Configuration File under 'Remap files'.

Step 3: Generate PRMS vegetation parameters from veg_parameters.py: The veg_parameters.py script will use the given LANDFIRE datasets and the associated remap files to write vegetation codes acceptable to PRMS to the parameter shapefile attribute table (figure 20). The cell size setting designates the resolution at which the data will be re-projected, so a factor of the fishnet grid is the most acceptable for best alignment (10m is default). Choosing a projection cell size that is not a multiple of the fishnet cell size could cause unwanted misalignment of the dataset. The script will scan the vegetation rasters to check that all values are in the remap files, and will alert the user of missing vegetation codes, if any. The remapped files will be stored in a folder named 'veg_rasters' in the working directory that will appear once the script has run. To evaluate these files, open the rasters and their attribute tables in ArcMap. Also check the HRU parameter shapefile to ensure the vegetation related fields were populated for all active cells. Vegetation fields include COV_TYPE, COVDEN_SUM, COVDEN_WIN, RAD_TRNCF, SNOW_INTCP, SRAIN_INTC, and WRAIN_INTC.

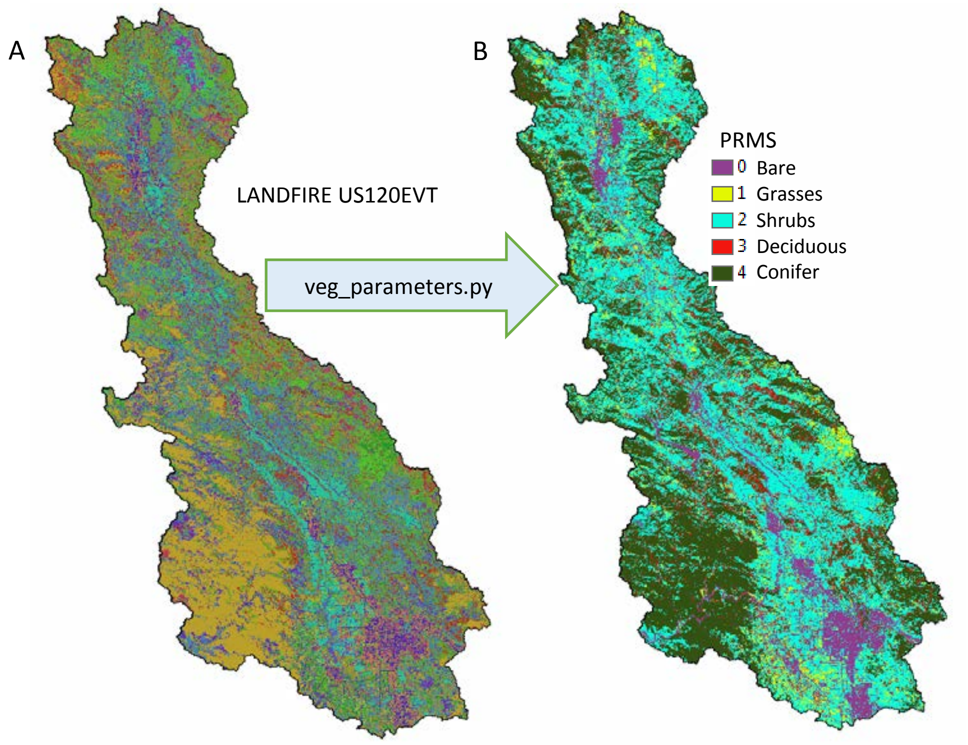
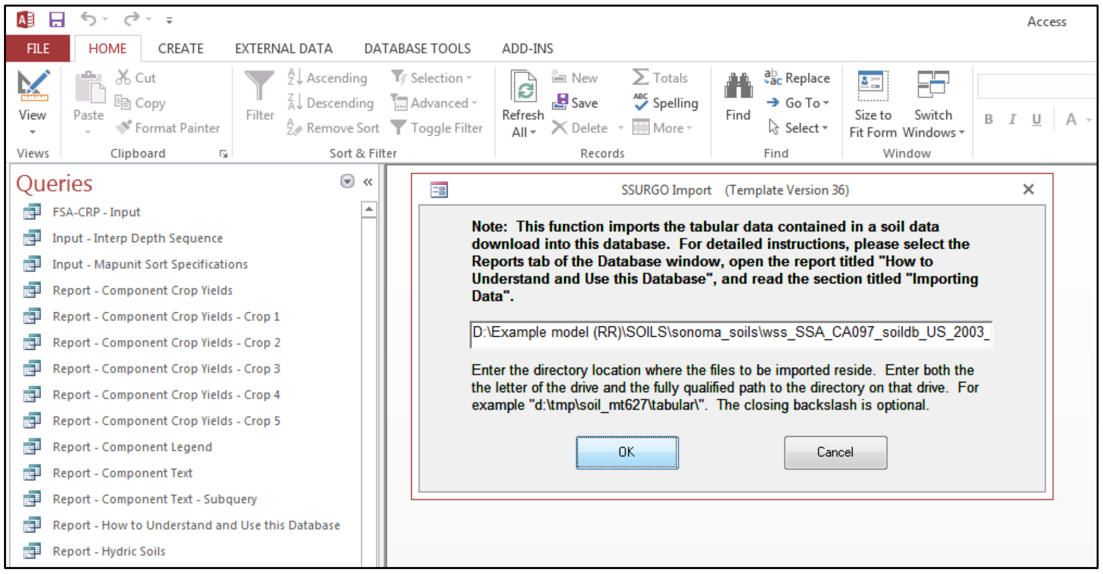


Figure 20: Vegetation type remapped from A) LANDFIRE existing vegetation type to B) PRMS values.

II. Generate PRMS soilzone parameters using Soil Survey Spatial and Tabular Data (SSURGO), and/or the U.S. General Soil Map (STATSGO2).

Some preparation of the soil data is necessary for Gsflow-Arcpy. The required soil data must be imported through the ArcMap Soil Data Viewer add-in, then formatted as raster images. The reason the Soil Data Viewer is listed as a recommended software is because the user could choose to simply use PRMS default values for soil parameters. This is not recommended, but is an option for a quick model build. Detailed instructions for installing and using the Soil Data Viewer are available on the USDA web page (www.nrcs.usda.gov/wps/portal/nrcs/detail/soils/survey/geo/?cid=nrcseprd337066). The following steps take the user through the process of downloading, importing, mapping, and formatting soil data.

Step 1: Prepare soil data for ArcMap: Soil data can be downloaded through the Geospatial Data Gateway. For the Russian River watershed, SSURGO data was used for Mendocino and Sonoma counties, as it completely covered the study area. In cases where SSURGO data does not completely cover the model domain, STATSGO data must be downloaded and can be used in conjunction with the SSURGO data. The data for Mendocino County downloaded in three separate folders, while Sonoma county data was contained in a single folder. Zipped folders must be extracted and the Microsoft Access files that come with the data must be opened in order to import the soil data needed for PRMS. Upon opening the Microsoft Access file for each database, a window labeled “Macro Single Step” may appear. Select ‘Stop All Macros’ and then Enable Content. The user will then be prompted to enter the directory (file pathname) where the soil data is stored. Paste the directory for the tabular data into the SSURGO Import window and select OK to import the data (figure 21). Once imported, the data will be available for the NRCS Soil Data Viewer Tool add-in in ArcMap.


Access
FILE
HOME
CREATE
EXTERNAL DATA
DATABASE TOOLS
ADD-INS
View
Paste
Cut
Copy
Format Painter
Filter
Ascending
Descending
Remove Sort
Selection
Advanced
Toggle Filter
Refresh All
New
Save
Delete
Totals
Spelling
More
Find
Replace
Go To
Select
Size to Fit Form
Switch Windows
Views
Clipboard
Sort & Filter
Records
Find
Window
Queries
FSA-CRP - Input
Input - Interp Depth Sequence
Input - Mapunit Sort Specifications
Report - Component Crop Yields
Report - Component Crop Yields - Crop 1
Report - Component Crop Yields - Crop 2
Report - Component Crop Yields - Crop 3
Report - Component Crop Yields - Crop 4
Report - Component Crop Yields - Crop 5
Report - Component Legend
Report - Component Text
Report - Component Text - Subquery
Report - How to Understand and Use this Database
Report - Hydric Soils
SSURGO Import (Template Version 36)
OK
Cancel

Note: This function imports the tabular data contained in a soil data download into this database. For detailed instructions, please select the Reports tab of the Database window, open the report titled "How to Understand and Use this Database", and read the section titled "Importing Data".

D:\Example model (RR)\SOILS\sonoma_soils\wss_SSA_CA097_soildb_US_2003_

Enter the directory location where the files to be imported reside. Enter both the the letter of the drive and the fully qualified path to the directory on that drive. For example "d:\tmp\soil_mt627\tabular\". The closing backslash is optional.

Figure 21: Importing SSURGO data into Microsoft Access. Once hitting OK, the data can take several minutes to import.

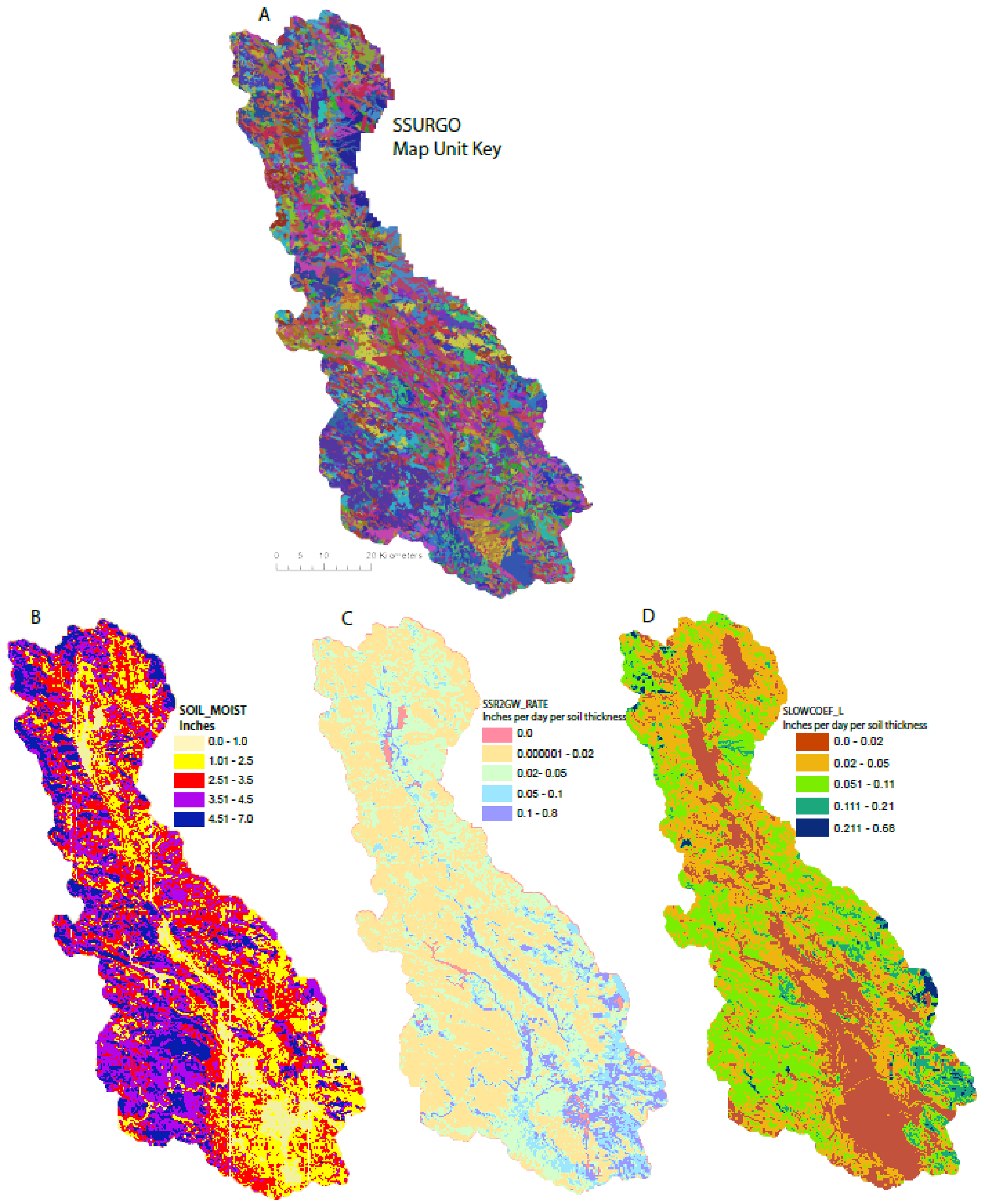
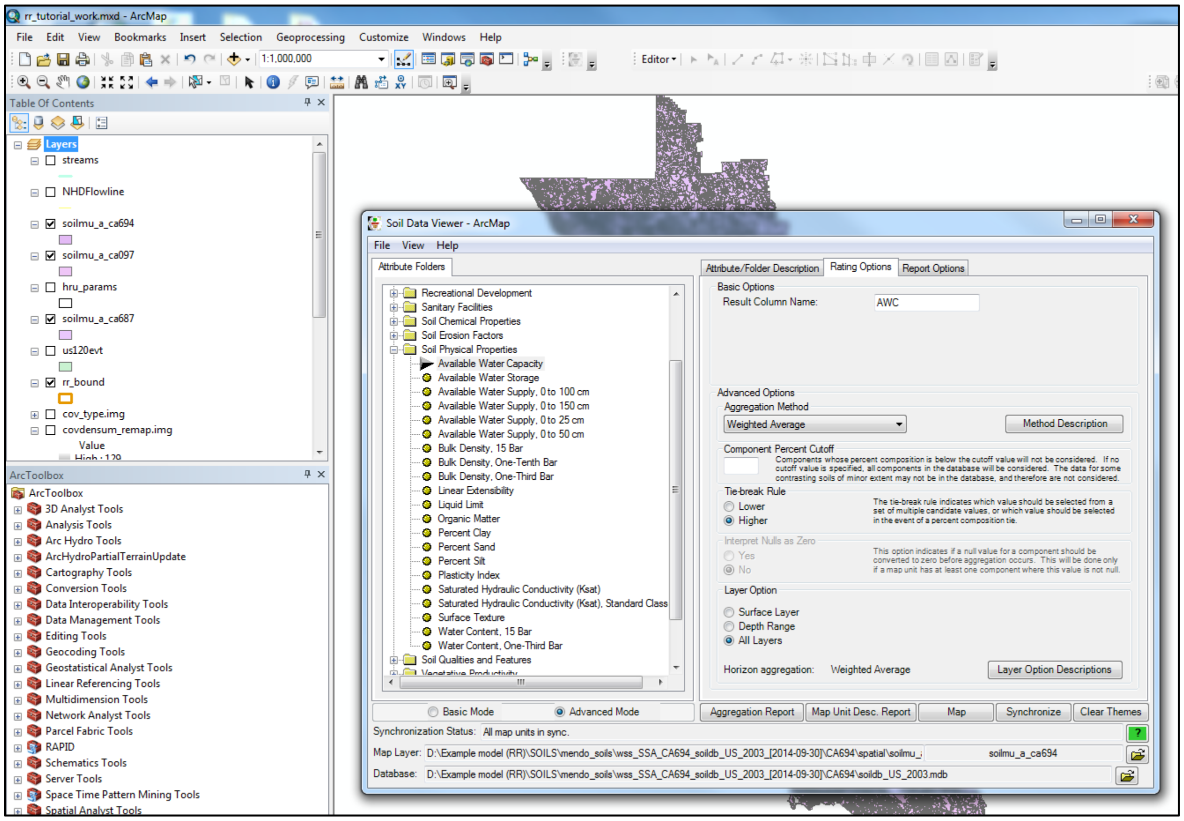


Figure 22: A visualization of how soil data is transferred from A) the mosaic of the SSURGO map unit keys for the Russian River watershed and resulting rasters of soil properties; B) MOIST_MAX (PRMS parameter soil_moist_max); C) SSR2GW_RATE (PRMS parameter ssr2gw_rate); and D) SLOWCOEF_L (PRMS parameter slow_coef_lin).

Step 2: Load soil property maps into ArcMap: With the SSURGO/STATSGO soil data shapefiles (located in the spatial folder) open in ArcMap, open the Soil Data Viewer tool (an icon should appear on the ArcMap toolbar after importing the add-in). A box will appear for the user to choose which soil dataset will be used, and then the main Soil Data

Viewer window will open. Paste the Microsoft Access pathname of the matching file in the Database box (figure 23). These two files (Map Layer and Database) should synchronize, giving the green light to download specific attributes to ArcMap. Once the database file path has been entered into the Soil Data Viewer, the user will see all related attribute folders associated with the database. PRMS is interested in Available Water Capacity (AWC), Percent Clay, Percent Sand, Percent Silt, and Saturated Hydraulic Conductivity (Ksat), and soil depth. These parameters are located in the Soil Physical Properties folder when the 'Advanced Mode' option is selected, except soil depth, which is located in the Soil Qualities and Features folder. Each parameter must be selected and rating options chosen before mapping the data to ArcMap. Soil depth is optional since Gsflow-Arcpy will apply the greatest value of root depth vs soil depth.



Basic Options

Result Column Name: AWC

Advanced Options

Aggregation Method: Weighted Average

Component Percent Cutoff

Components whose percent composition is below the cutoff value will not be considered. If no cutoff value is specified, all components in the database will be considered. The data for some contrasting soils of minor extent may not be in the database, and therefore are not considered.

Tie-break Rule: Lower [✓ Higher]

The tie-break rule indicates which value should be selected from a set of multiple candidate values, or which value should be selected in the event of a percent composition tie.

Interpret Nulls as Zero: Yes [✓ No]

This option indicates if a null value for a component should be converted to zero before aggregation occurs. This will be done only if a map unit has at least one component where this value is not null.

Layer Option: Surface Layer, Depth Range, [✓ All Layers]

Horizon aggregation: Weighted Average

Basic Mode [✓ Advanced Mode]

Synchronization Status: All map units in sync.

Map Layer: D:\Example model (RR)\SOILS\mendo_soils\wss_SSA_CA694_soildb_US_2003_[2014-09-30]\CA694\spatial\soilmu_; soilmu_a_ca694

Database: D:\Example model (RR)\SOILS\mendo_soils\wss_SSA_CA694_soildb_US_2003_[2014-09-30]\CA694\soildb_US_2003.mdb

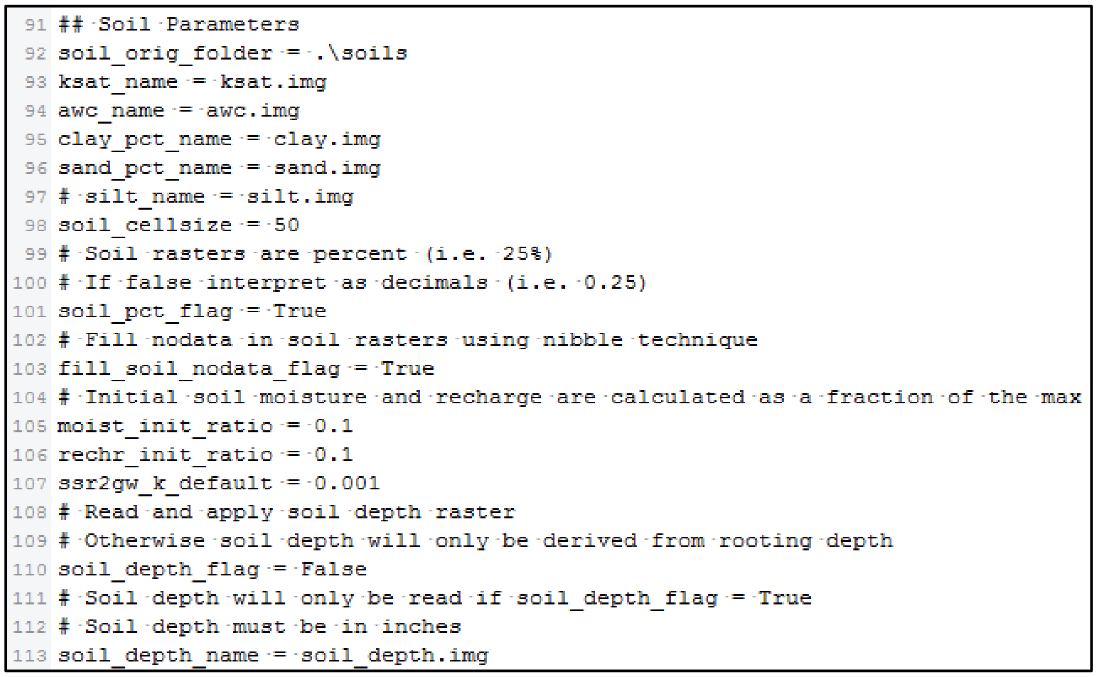
Figure 23: The Soil Data Viewer add-in open in ArcMap. Map Layer and Database file pathnames at the bottom of the window are synchronized. The drop-down menu below the Soil Physical Properties attribute contains the soil properties needed for PRMS. Rating options must be selected prior to clicking the 'Map' button.

Regarding the rating options, the 'Weighted Average' aggregation method is typically used, and 'Higher' is selected for the tie break rule ('Fastest' for Ksat). For the Layer Option, a depth range can be determined or 'All Layers' may be chosen. These settings do not seem to pose significant differences in the mapping of the soil properties relating to PRMS, so the user may feel free to experiment here and perform outside research to come up with the best options for a particular property. After filling out these options, press the ‘Map’ button to create a shapefile containing the selected soil property. This step needs to be done for each map layer (soil shapefile) and each soil property.

Step 3: Create raster layers for each soil property: Shapefiles containing soil properties may only cover partial sections of the model domain, especially if the study area crosses county lines. Thus, the shapefiles must be joined to form one single shapefile (per soil property) covering the entire study area. These shapefiles must then be converted to raster images in order to be read correctly by Gsflow-Arcpy. The final result should be raster image files covering the study area for each soil property. For more information on these ArcGIS methods, refer to ESRI online resources: http://resources.arcgis.com/en/help/.

A surficial geology raster can also be created, in order to apply a more realistic saturated hydraulic conductivity (Ksat) to the soil gravity drainage calculations. This raster is optional, and a default multiplier value can be set in the Configuration File if the user does not choose to use a geology raster. The rock type from a surficial geology map is used to apply a multiplier to the Ksat acquired from the SSURGO data. It has been observed that often these conductivity values are too large and cause parameters such as ssr2gw_rate to be much too high initially. Geology maps that are GIS ready can be downloaded at https://mrdata.usgs.gov/geology/state/. The user must then select (and possibly merge) the polygons covering the model domain, and convert to a raster, just like the soil data. A VALUE field must be added to the attribute table that contains a multiplier for each rock type. This optional step can help designate recharge zones, for example, that may not have been as well defined by the soil properties alone.

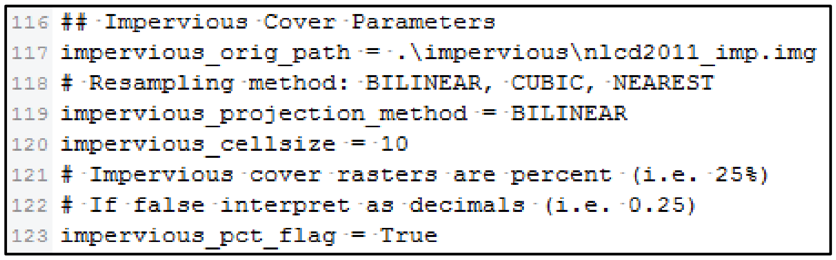
Step 4: Run soil_raster_prep.py and soil_parameters.py: Ensure that the correct file-names for the soil rasters have been entered in the Configuration File under the ‘Soil parameters’ section (figure 24) , and run soil_raster_prep.py. This script evaluates each soil raster, projecting and clipping to the HRU parameter grid and filling ‘no data’ areas using Nibble techniques (spatial analyst tool in Arc). Once the soil prep script has finished, run soil_parameters.py. This will populate the soil property fields in the parameter shapefile attribute table (AWC, CLAY_PCT, SAND_PCT, KSAT, ROOT_DEPTH, SOIL_TYPE, MOIST_INIT, MOIST_MAX, RECHR_INIT, RECHR_MAX, SSR2GW_RATE, SLOWCOEF_L, and SLOWCOEF_S). The soil files will be stored in the ‘soil_rasters’ folder in the working directory, allowing for the user to observe the image files in ArcMap.



```
## Soil Parameters
soil_orig_folder = .\soils
ksat_name = ksat.img
awc_name = awc.img
clay_pct_name = clay.img
sand_pct_name = sand.img
# silt_name = silt.img
soil_cellsize = 50
# Soil rasters are percent (i.e. 25%)
# If false interpret as decimals (i.e. 0.25)
soil_pct_flag = True
# Fill nodata in soil rasters using nibble technique
fill_soil_nodata_flag = True
# Initial soil moisture and recharge are calculated as a fraction of the max
moist_init_ratio = 0.1
rechr_init_ratio = 0.1
ssr2gw_k_default = 0.001
# Read and apply soil depth raster
# Otherwise soil depth will only be derived from rooting depth
soil_depth_flag = False
# Soil depth will only be read if soil_depth_flag = True
# Soil depth must be in inches
soil_depth_name = soil_depth.img
```

Figure 24: Soil parameters section of the Configuration File. The folder path containing the soil data is designated, and each raster image filename is set for each soil property.

Step 5: Run impervious_parameters.py: The impervious cover data from the National Land Cover Database is applied to the parameter shapefile in this script, as an impervious percentage within each HRU grid cell. No formatting or altering is necessary for this data. Ensure that the folder in which the impervious database is stored is entered in the Configuration File under ‘Impervious cover parameters’ (figure 25) and run impervious_parameters.py. Open the parameter shapefile once the script finishes and observe the results by checking the attribute table and/or using ‘Symbology’ properties to visualize results.



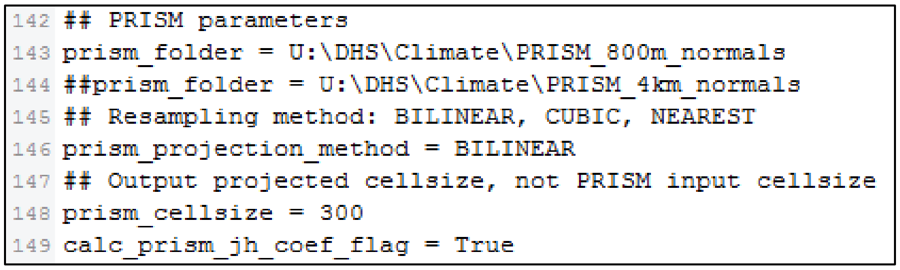
```
## Impervious Cover Parameters
impervious_orig_path = .\impervious\nlcd2011_imp.img
# Resampling method: BILINEAR, CUBIC, NEAREST
impervious_projection_method = BILINEAR
impervious_cellsize = 10
# Impervious cover rasters are percent (i.e. 25%)
# If false interpret as decimals (i.e. 0.25)
impervious_pct_flag = True
```

Figure 25: Impervious coverage parameters in the Configuration File. Cell size is the re-projection size, once again best if a multiple of the grid cell size is used.

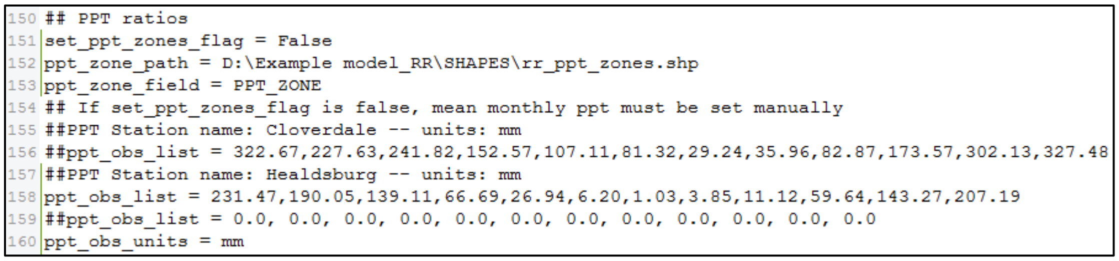
III. Set temperature and precipitation parameters using observed station data and PRISM climate data.

PRMS parameters used for distributing temperature and precipitation to HRUs are generated using data collected at select climate station(s) within or near the study area along with PRISM gridded climate data. Specific data is needed depending on the climate modules being applied to the model. See the PRMS documentation for more information on the specific parameters needed for each climate module. The following steps explain the process and calculations that result in each HRU receiving a unique monthly maximum and minimum temperature and a monthly precipitation amount along with a precipitation factor. These data will appear in the parameter shapefile attribute table and are written to the PRMS Parameter File to be applied with the several possible modules available. As PRMS is driven by daily climate data from one or two climate stations (point data), the precipitation multipliers, or ratios, derived through the following steps allow for a more realistic distribution of climate data over the model domain.

Step 1: Prepare PRISM and climate station data: PRISM data can be downloaded through the link provided at the beginning of this manuscript. The options available for data download include different scales and time periods. Typically, and for this tutorial, the 30 year 800m monthly normals are used; however, the user may choose to use different PRISM data. Longer periods are available for purchase. The downloaded climate data must be saved into separate folders containing the precipitation (ppt), maximum temperature (tmax), and minimum temperature (tmin). In the Configuration File, enter the folder path for the PRISM data in the 'PRISM parameters' section (figure 25).


```
## PRISM parameters
prism_folder = U:\DHS\Climate\PRISM_800m_normals
##prism_folder = U:\DHS\Climate\PRISM_4km_normals
## Resampling method: BILINEAR, CUBIC, NEAREST
prism_projection_method = BILINEAR
## Output projected cellsize, not PRISM input cellsize
prism_cellsize = 300
calc_prism_jh_coef_flag = True
```
Figure 25: PRISM parameter settings in the Configuration File. PRISM cell size is the output cell size desired (cell size, or multiple, of the model grid).

To distribute precipitation data over the model domain, mean monthly precipitation values need to be determined from observed data. These are values calculated from a selected climate station or stations, typically the same station(s) that will be driving the model with daily values. If a single station is being used, the observed mean monthly values are entered in a comma separated row in the 'PPT ratios' section of the Configuration File for *ppt_obs_list* (figure 26).



```
## PPT ratios
set_ppt_zones_flag = False
ppt_zone_path = D:\Example model_RR\SHAPES\rr_ppt_zones.shp
ppt_zone_field = PPT_ZONE
## If set_ppt_zones_flag is false, mean monthly ppt must be set manually
##PPT Station name: Cloverdale -- units: mm
##ppt_obs_list = 322.67,227.63,241.82,152.57,107.11,81.32,29.24,35.96,82.87,173.57,302.13,327.48
##PPT Station name: Healdsburg -- units: mm
ppt_obs_list = 231.47,190.05,139.11,66.69,26.94,6.20,1.03,3.85,11.12,59.64,143.27,207.19
##ppt_obs_list = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
ppt_obs_units = mm
```

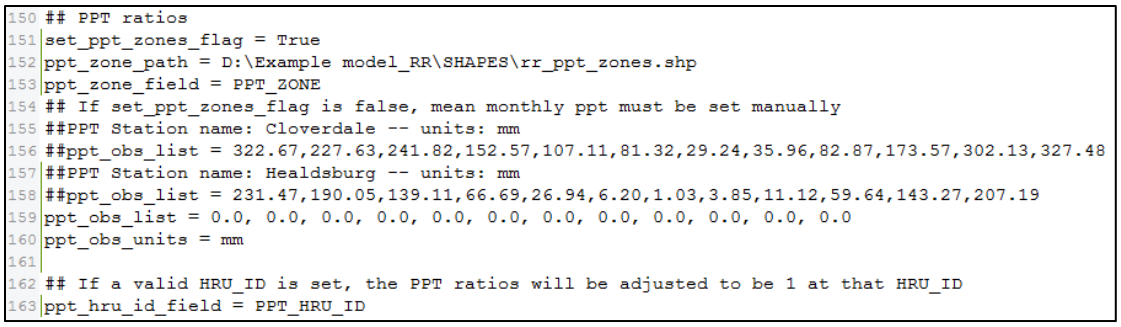
Figure 26: For use of a single climate station, set_ppt_zones_flag is set to false, and ppt_obs_list is set equal to monthly mean precipitation values from a climate station. Two climate stations are shown in this example, but only one is read by the script (Healdsburg), while the other (Cloverdale) is commented out by the '##'. Units of the observed data are also set here.

A single climate station can be used for simplicity or for smaller models, but an option exists for more than one climate station to be used. The Russian River is a large watershed and contains several climate stations with decent records. A handful NWS-COOP stations were chosen to distribute precipitation and temperature over the model domain (figure 27). Data were obtained through the Western Regional Climate Center (WRCC) climate summaries: http://www.wrcc.dri.edu/climate-summaries/. From this link, the NOAA cooperative stations interactive map can be accessed to locate each station and its data.

network	sta_id	sta_name	record_start	record_end	Lat	lon	elev_ft
COOP	049684	WILLITS 1NE	3/6/1902	9/27/2012	39.41944	-123.34	1353
COOP	047109	POTTER VALLEY	9/1/1937	6/9/2016	39.36194	-123.12	1018
COOP	049122	UKIAH	1/1/1893	5/24/2013	39.14667	-123.21	636
COOP	049173	UPPER LAKE	1/1/1893	11/12/2006	39.19056	-122.89	1335
COOP	044701	LAKEPORT	1/1/1920	6/20/2002	39.03333	-122.91	1315
COOP	041838	CLOVERDALE	7/22/1950	5/31/2016	38.8	-123.01	330
COOP	049440	WARM SPRINGS DAM	6/1/1973	1/3/2006	38.71611	-122.99	224
COOP	043875	HEALDSBURG	2/1/1893	8/31/2012	38.6175	-122.87	108
COOP	041312	CALISTOGA	3/1/1906	5/20/2016	38.59611	-122.60	400
COOP	047971	SANTA ROSA SONOMA CO AP	6/8/1998	6/9/2016	38.50389	-122.81	114
COOP	043578	GRATON	1/1/1926	4/30/2016	38.43056	-122.86	200
COOP	046370	OCCIDENTAL	5/1/1943	5/26/2016	38.38583	-122.96	865
COOP	047965	SANTA ROSA	6/1/1902	1/31/2013	38.43806	-122.69	174

Figure 27: NWS-COOP stations chosen to represent climate over the Russian River model domain

In the case that multiple stations will be used to distribute climate data over a large model domain, zones must be designated to assign the HRUs to a specified climate station. To do this, a shapefile needs to be created that contains the chosen precipitation zone polygons and holds the mean monthly precipitation values for each station (PPT_01, PPT_02 etc.) associated with a precipitation zone. The user must set *set_ppt_zones_flag* to True in the Configuration File, and provide the location and name of the precipitation zone shapefile, along with the field name of the zone field (figure 28).



```
## PPT ratios
set_ppt_zones_flag = True
ppt_zone_path = D:\Example model_RR\SHAPES\rr_ppt_zones.shp
ppt_zone_field = PPT_ZONE
## If set_ppt_zones_flag is false, mean monthly ppt must be set manually
##PPT Station name: Cloverdale -- units: mm
##ppt_obs_list = 322.67,227.63,241.82,152.57,107.11,81.32,29.24,35.96,82.87,173.57,302.13,327.48
##PPT Station name: Healdsburg -- units: mm
##ppt_obs_list = 231.47,190.05,139.11,66.69,26.94,6.20,1.03,3.85,11.12,59.64,143.27,207.19
ppt_obs_list = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
ppt_obs_units = mm

## If a valid HRU_ID is set, the PPT ratios will be adjusted to be 1 at that HRU_ID
ppt_hru_id_field = PPT_HRU_ID
```

Figure 28: PPT Ratios section when using more than one climate station. Zone flag is set to True, observed data is commented out, and *ppt_zone_path*, *ppt_zone_field*, and *ppt_hru_id_field* are specified. Observed units are still designated here.

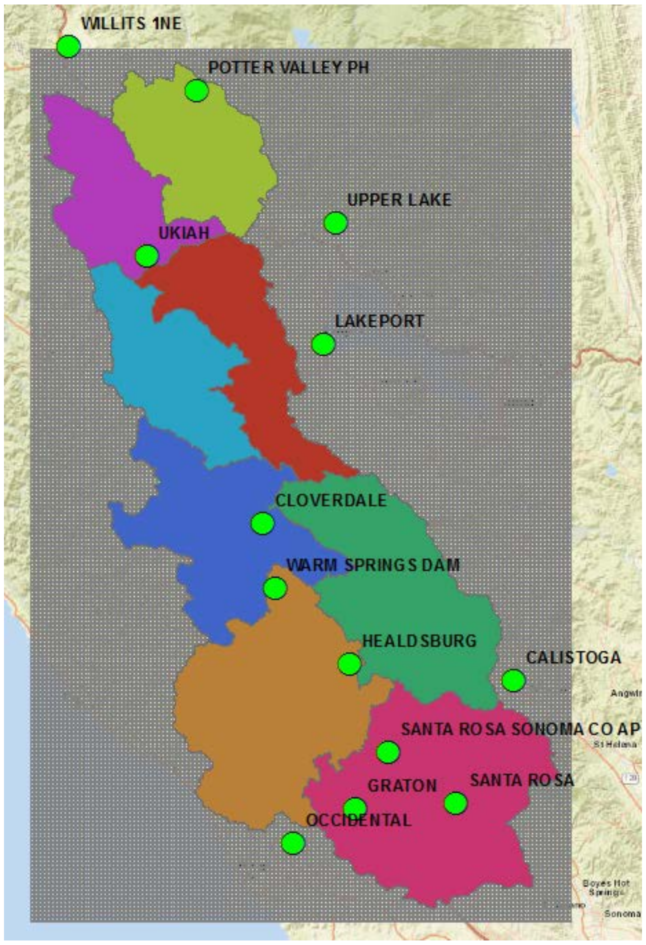
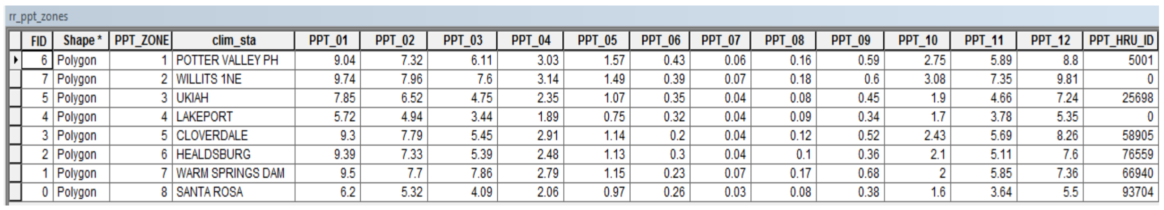


Figure 29: Precipitation zone polygons and NWS-COOP climate stations. Precipitation distribution will be based on the climate station assigned to each zone, and the relationship of the mean monthly precipitation of that station to the PRISM data.

To assign the precipitation zones for the Russian River model, HUC 12 polygons were merged together to designate 8 zones (figure 29). A climate station was chosen to represent each zone, and the mean monthly precipitation was input to the attribute table of the precipitation zone shapefile (figure 30). The HRU_ID where each station is located can be entered into the PPT_HRU_ID field to ensure the precipitation ratio at that HRU is equal to 1. If the station is outside the zone or the model domain, a zero entered will keep the script from attempting to adjust the precipitation ratio to 1 at the station location. The final PPT zone shapefile should contain only as many rows as PPT zones. It can help to create the zone shapefile directly from the HRU parameter shapefile, by extracting the data to a new shapefile and editing the attribute table. This will ensure no gaps are in the precipitation zones and they will snap to the model grid. When selecting the PPT zones, be sure to make the zone shapefile the only selectable layer so that the selected cells can be merged into one polygon. Also make sure that the twelve (monthly) PPT columns are type float.


rr_ppt_zones

FID	Shape *	PPT_ZONE	clim_sta	PPT_01	PPT_02	PPT_03	PPT_04	PPT_05	PPT_06	PPT_07	PPT_08	PPT_09	PPT_10	PPT_11	PPT_12	PPT_HRU_ID
6	Polygon	1	POTTER VALLEY PH	9.04	7.32	6.11	3.03	1.57	0.43	0.06	0.16	0.59	2.75	5.89	8.8	5001
7	Polygon	2	WILLITS 1NE	9.74	7.96	7.6	3.14	1.49	0.39	0.07	0.18	0.6	3.08	7.35	9.81	0
5	Polygon	3	UKIAH	7.85	6.52	4.75	2.35	1.07	0.35	0.04	0.08	0.45	1.9	4.66	7.24	25698
4	Polygon	4	LAKEPORT	5.72	4.94	3.44	1.89	0.75	0.32	0.04	0.09	0.34	1.7	3.78	5.35	0
3	Polygon	5	CLOVERDALE	9.3	7.79	5.45	2.91	1.14	0.2	0.04	0.12	0.52	2.43	5.69	8.26	58905
2	Polygon	6	HEALDSBURG	9.39	7.33	5.39	2.48	1.13	0.3	0.04	0.1	0.36	2.1	5.11	7.6	76559
1	Polygon	7	WARM SPRINGS DAM	9.5	7.7	7.86	2.79	1.15	0.23	0.07	0.17	0.68	2	5.85	7.36	66940
0	Polygon	8	SANTA ROSA	6.2	5.32	4.09	2.06	0.97	0.26	0.03	0.08	0.38	1.6	3.64	5.5	93704

Figure 30: Example of a PPT zone shapefile attribute table. Eight zones were designated for the Russian River tutorial, and mean monthly observed precipitation values from each associated station were entered. The ratio of the observed station values to the PRISM data will determine the precipitation distribution within each precipitation zone. This option allows for a more realistic precipitation distribution over larger model domains.

Step 2: Run prism_800m_normals.py and ppt_ratio_parameters.py: These scripts will populate the temperature and precipitation fields in the parameter shapefile attribute table. The first script, prism_800m_normals.py (or prism_4km_normals, if the user is applying 4km PRISM data), reads the PRISM data from the three folders (ppt, tmax, tmin) and re-samples the temperature and precipitation data to the HRU grid, for each month of the year. The second script, ppt_ratio_parameters.py, uses the observed mean monthly precipitation values to calculate a ratio between the PRISM precipitation and the observed data, and assigns each HRU a monthly precipitation factor that acts as a multiplier in order to account for changes in elevation (and effects on precipitation) throughout the model domain. Observe the results in the attribute table in ArcMap, and make sure the precipitation factors make sense (figure 31). For example, in a mountainous region, the ratios would mostly be <1 if a high elevation climate station was used, and >1 if a low elevation station was used. These factors influence the rain and snow adjust parameters in the PRMS Parameter File.

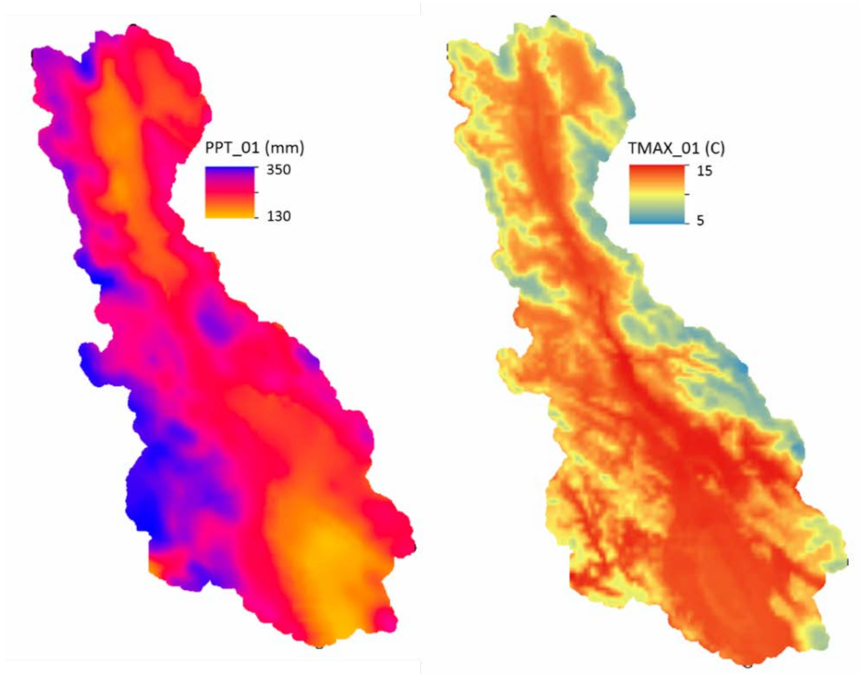


Figure 31: Precipitation and maximum temperature distribution for the month of January, based on observed climate data and the PRISM network. (note: these distributions are based off a single climate dataset)

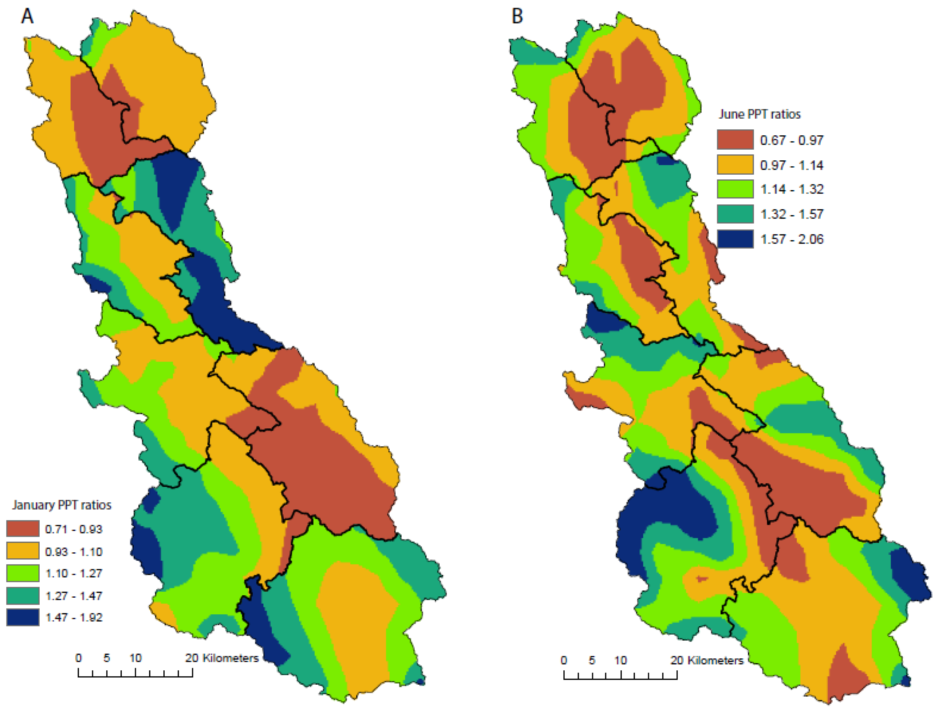
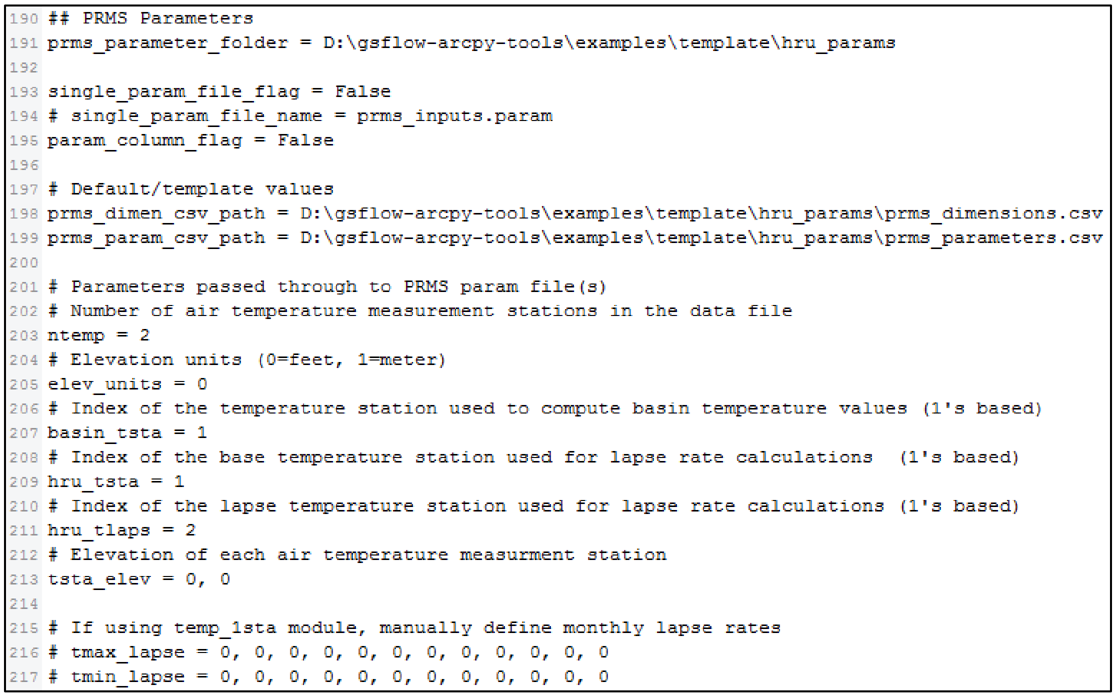


Figure 32: Precipitation factors for January are calculated from January PRISM values and daily precipitation values measured at climate stations located within the watershed. Nine precipitation stations and respective zones were used in this example. This process is repeated for each month.

IV. Finalize and build PRMS Parameter File

At this point, all PRMS related scripts have been run and the attribute table in the parameter shapefile will be fully populated and prepared to develop the Parameter File needed to run PRMS, along with several input files for MODFLOW. A couple final steps will ensure successful development of the Parameter File.

Step 1: Designate Parameter File name, and CSV file locations: Review the PRMS Parameters section of the Configuration File, and make sure to have the file path names entered correctly, along with several model specific settings for the parameter and dimension CSV files (figure 33). The parameter CSV file holds default values for some parameters as well parameters that will be designated from the HRU shapefile and the Configuration File. Look over these settings and compare the default values to those given in the PRMS documentation. The user may want to adjust the defaults within the possible range of values depending on the study area. Typically, however, the default values are left alone until the calibration period, where they may be changed in the actual Parameter File. The dimension CSV file contains counts of certain parameters, some of which are specific to a study area, such as number of precipitation stations (nrain), temperature stations (ntemp), and streamgage stations (nobs). These two CSV files must be present in the working directory for the final prms_template_fill.py script to run. Temperature station settings in this section of the Configuration File must reflect the temperature module that will be used. These parameters are explained in the PRMS documentation. Depending on the version of PRMS being used, the user may choose to have Gsflow-Arcpy make a single Parameter File with parameters listed in a single column (applicable to versions through PRMS 4) or multiple Parameter Files with parameters in 2D arrays (applicable to PRMS 5). These settings are easily designated by the flags in the Configuration File under the ‘PRMS Parameters’ section (figure 33).



```
190 ## PRMS Parameters
191 prms_parameter_folder = D:\gsflow-arcpy-tools\examples\template\hru_params
192
193 single_param_file_flag = False
194 # single_param_file_name = prms_inputs.param
195 param_column_flag = False
196
197 # Default/template values
198 prms_dimen_csv_path = D:\gsflow-arcpy-tools\examples\template\hru_params\prms_dimensions.csv
199 prms_param_csv_path = D:\gsflow-arcpy-tools\examples\template\hru_params\prms_parameters.csv
200
201 # Parameters passed through to PRMS param file(s)
202 # Number of air temperature measurement stations in the data file
203 ntemp = 2
204 # Elevation units (0=feet, 1=meter)
205 elev_units = 0
206 # Index of the temperature station used to compute basin temperature values (1's based)
207 basin_tsta = 1
208 # Index of the base temperature station used for lapse rate calculations  (1's based)
209 hru_tsta = 1
210 # Index of the lapse temperature station used for lapse rate calculations (1's based)
211 hru_tlaps = 2
212 # Elevation of each air temperature measurment station
213 tsta_elev = 0, 0
214
215 # If using temp_1sta module, manually define monthly lapse rates
216 # tmax_lapse = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
217 # tmin_lapse = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
```

Figure 33: Folder paths and file names for the PRMS Parameter File and accompanying CSV files in the Configuration File. Temperature and elevation settings must be set based on the module being used (temp_1sta vs temp_laps). This will provide PRMS with adequate information to distribute temperature over the model domain.

Step 2. Run prms_tmeplate_fill.py: This script will create the initial Parameter File(s) that PRMS will read. The Parameter File option (lines 193-195 in figure 33) directs the script to create a single PRMS Parameter File or multiple Parameter Files (categorized in the parameters.csv file), and the option to create parameter values in arrays or columns. Default (false) settings will create multiple Parameter Files with arrays. After choosing the settings and running the script, review the PRMS Parameter File(s) and ensure that the initial build contains reasonable parameter values, and correct units are assigned to match the data. It is easy to forget about maintaining consistent units while developing the model data. Units used in the PRMS Data File to drive the model must match those of the Parameter File. These include temperature, precipitation, and runoff. For example, the units for the precipitation factors used in this tutorial are millimeters. This means that the precipitation data in the Data File must also be in millimeters. This is the final step for this version of Gsflow-Arcpy. To run PRMS, the user must also create a Data File and a Control File. Guidelines for making these files can be found in the PRMS documentation.

For MODFLOW models, several columns in the HRU shapefile attribute table are ready to be formatted into MODFLOW input files. Resources are available that can automate this procedure somewhat, such as the MODFLOW FlowPy toolkit (https://github.com/modflowpy/flopy). Layer bottom elevations can quickly be calculated based on DEM_ADJ, and SFR package inputs are in the attribute table. A GSFLOW model can be created once both PRMS and MODFLOW models have been separately constructed and calibrated.

References

Viger, R.J., and Leavesley, G.H., 2007, The GIS Weasel User's Manual: U.S. Geological Survey Techniques and Methods, book 6, chap. B4, 201 p; http://pubs.usgs.gov/tm/2007/06B04/

Markstrom, S.L., Regan, R.S., Hay, L.E., Viger, R.J., Webb, R.M.T., Payn, R.A., and LaFontaine, J.H., 2015, PRMS-IV, the precipitation-runoff modeling system, version 4: U.S. Geological Survey Techniques and Methods, book 6, chap. B7, 158 p., http://dx.doi.org/10.3133/tm6B7.

Henson, W.R., Medina, R.L., Mayers, C.J., Niswonger, R.G., and Regan, R.S., 2013, CRT—Cascade Routing Tool to define and visualize flow paths for grid-based watershed models: U.S. Geological Survey Techniques and Methods 6-D2, 28 p.

Maidment, D.R., 2002, Arc Hydro: GIS for Water Resources, vol 1, ESRI, Inc. 203 p.

Regan, R.S., Markstrom, S.L., Hay, L.E., Viger, R.J., Norton, P.A., Driscoll, J.M., LaFontaine, J.H., 2017, Description of the National Hydrologic Model for use with the Precipitation-Runoff Modeling System (PRMS): U.S. Geological Survey Techniques and Methods, book 6, chap B9, xxx p. DOI 10.3133