SeRFE

User Manual and Documentation

Jordan Gilbert

Contents

SeRFE version 0.1 User Manual

Software information and requirements

SeRFE users should have a basic familiarity with Python scripts and IDE software. Running the model does not require any coding, but it is currently run using Python scripts in an IDE. They should also be familiar with some sort of GIS software, which is used to interact with the spatial data.

Required software

Any GIS program (e.g. ArcGIS, QGIS)*
A Python Integrated Development Environment (IDE; e.g. Spyder PyCharm etc.)

Required Python installation:

Python 3

Required Python packages:

standard scientific python packages (numpy, scipy, matplotlib, scikit-learn)
GDAL
rasterio
fiona
geopandas
pandas
richdem**
shapely
rasterstats
PyQt5

*for interacting with geospatial data
**only required for producing a drainage area raster from a DEM in the Preprocessing tools

The SeRFE source code can be found at:
https://github.com/jtgilbert/SeRFE

To report issues or to contact the author, use this github repository.

1. **Overview**

The Sediment Routing and Floodplain Exchange (SeRFE) tool is a framework for spatially-explicit modeling of sediment routing and balance. Modeling is performed at the reach scale, but applied across entire watersheds, providing a comprehensive picture of the sediment dynamics within a basin. The primary function of SeRFE is to model the response in sediment storage and transfer to particular flow events, which can be real or hypothetical (simulated). River systems are affected by a broad range of impacts throughout watersheds, which can then propagate through drainage networks, resulting in changes downstream. This tool provides a means for assessing the magnitude and spatial extent of impacts to watersheds, which can provide important contextual information for river management and restoration.

SeRFE is run using a combination of widely available geospatial data, remotely sensed data and some field data. The basis of the model is a drainage network, which is segmented into a desired reach length. Various sub-models add attributes to this network which are then used to run the primary SeRFE model. The process for running SeRFE is illustrated in figure 1, and each of the modeling steps described in the following sections.

The model (all scripts) is contained within a directory. Inside this directory is a folder titled “data” that contains the .csv files that need to be filled out with input data (described below). This is the location where all additional data such as the downloaded geospatial data should be stored as well.

None of the actual modeling/processing occurs within a GIS program, however a GIS program is required to interact with the data and retrieve some information, which will be described later.

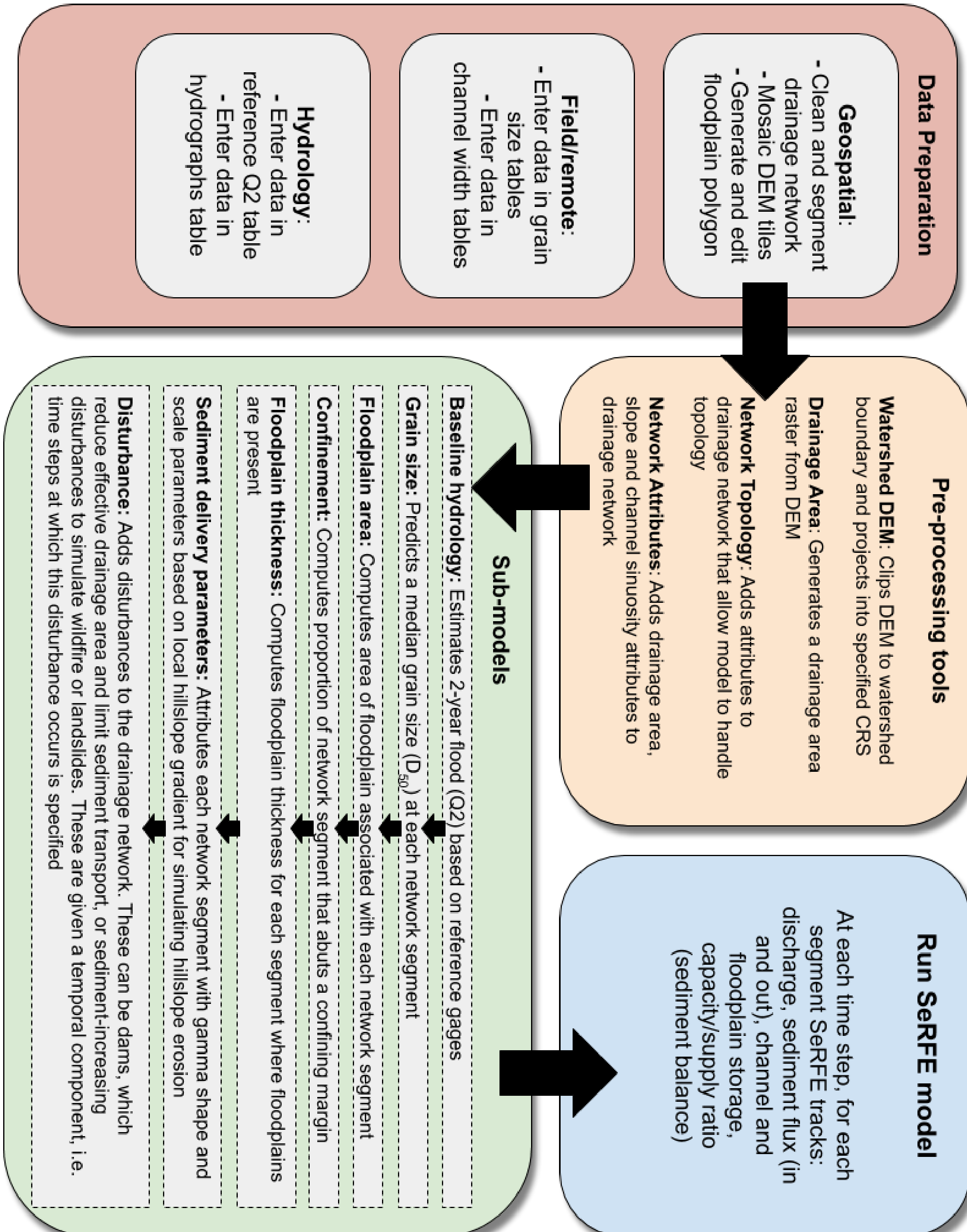

Figure 1. Workflow diagram for the SeRFE model.

2. Data Preparation

Geospatial data: SeRFE requires a drainage network and a digital elevation model (DEM). The best option for a drainage network is to extract one from the DEM being used using common GIS hydrologic tools (e.g. https://pro.arcgis.com/en/pro-app/tool-reference/spatial-analyst/identifying-stream-networks.htm). The National Hydrography Dataset (NHD) 24k drainage network, or similar existing datasets can be used, however the discrepancy between the stream network location and DEM network location will result in segments that are attributed with incorrect drainage area values which will need to be corrected manually. The National Elevation Dataset (NED) is a good option for a DEM. Both NHD and NED have national coverage and are freely available.

After obtaining a drainage network, it should be reduced to the desired portion of the network. For example, you could use a drainage area threshold of 50 km^2 to remove smaller, headwater streams, including all network segments whose contributing drainage area was >= this threshold. The network should then be dissolved into a single feature and subsequently segmented into a desired reach length, which is appropriate for the resolution of the DEM (e.g. we used ~500 - 1000 m segments with 10m DEMs).

For use in SeRFE, there can be NO gaps between segments of the drainage network. Occasionally there is no segment in the dataset to fill in a potential gap, in which case the user should manually digitize a segment to fill the gap, making sure that the segment is drawn starting at the upstream end and ending at the downstream end to ensure correct flow direction. Occasionally, a segment from the NHD network will have an incorrect flow direction (i.e. was drawn from downstream to upstream). These segments should be identified prior to use with the model, deleted and re-digitized with the correct flow direction (see https://desktop.arcgis.com/en/arcmap/10.3/manage-data/geometric-networks/displaying-flow-direction.htm). Incorrect flow directions cause problems with the network topology tools and will cause anomalies in the model outputs.

The NED dataset is generally delivered as individual tiles. If the watershed of interest consists of multiple tiles, the DEM tiles should be merged together prior to use with SeRFE. This can be accomplished, for example, using the “Mosaic to New Raster” tool in ArcGIS, or using the command line utilities that come with the “rasterio” python package required to run SeRFE (see https://rasterio.readthedocs.io/en/stable/cli.html#merge). If using ArcMap, in the tool’s environment settings make sure that the “raster analysis” resampling setting is set to “Bilinear” or “Cubic” to avoid the artifacts that “nearest neighbor” resampling introduces into DEMs.

The last required geospatial data is a valley bottom or floodplain polygon. We used the V-BET tool to accomplish this (Gilbert et al., 2016). V-BET is an ArcGIS tool https://github.com/Riverscapes/RCAT/releases, or an updated, open-source (non ArcGIS) tool https://github.com/jtgilbert/vbet-2. The user can also use any other method that they are familiar with for generating these polygons, or can manually digitize them.

All geospatial data should be projected into the same coordinate reference system (CRS; e.g. UTM). SeRFE's pre-processing tools provide a method for clipping and projecting DEMs (see below).

Interacting with the drainage network: At various points in the modeling process, scripts require a segment ID as an input for a particular stream segment. In ArcMap this is simply the "FID" of the segment and can be obtained by opening the attribute table, selecting the segment and noting the FID value. In QGIS, the "FID" field does not exist. Instead, select the segment of interest, open the attribute table, and hover the mouse pointer over the number at the left edge of the table. A segment ID will appear and can be noted (figure 2).

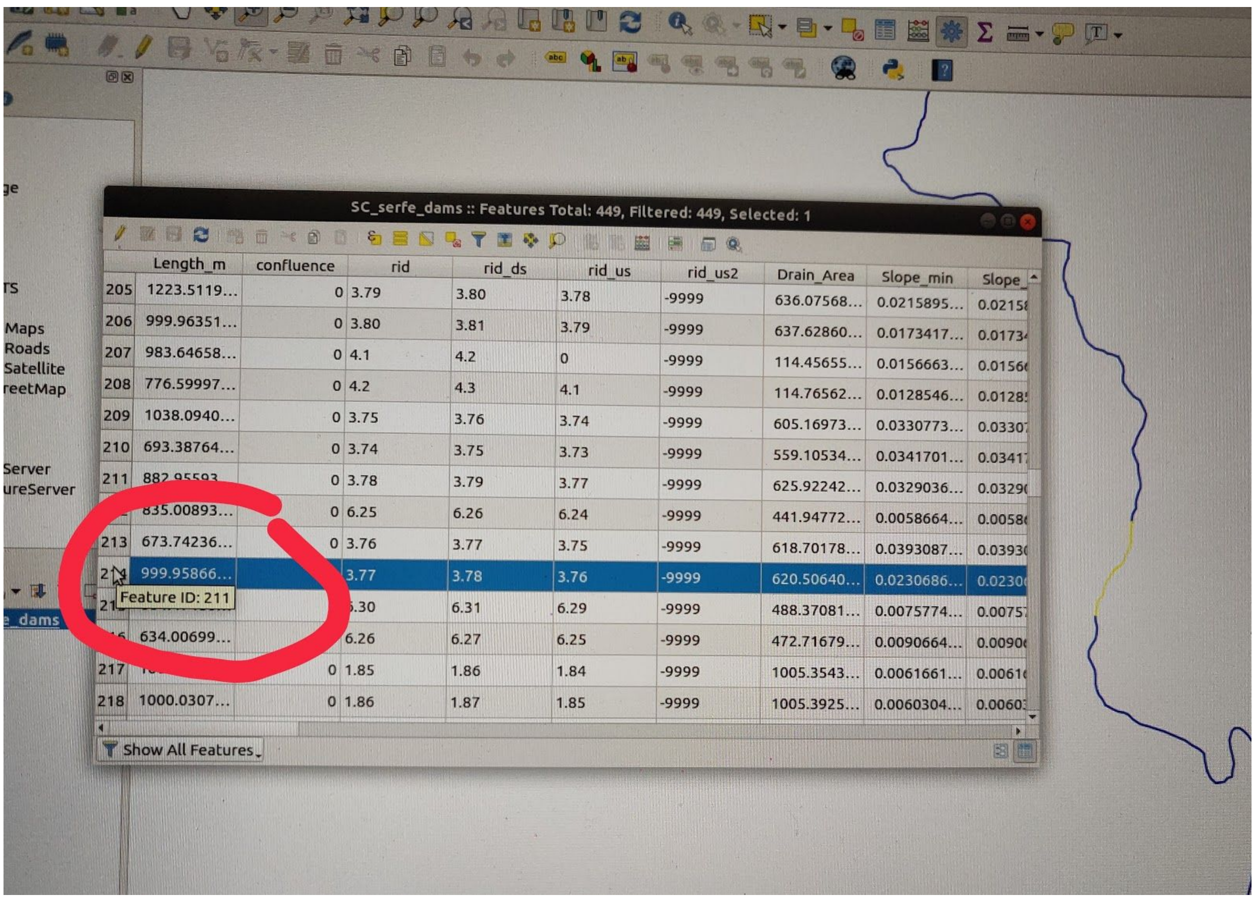

Figure 2. Identifying a segment ID number (211) in QGIS.

Field/Remote data: SeRFE uses field data on grain size distributions and channel widths to extrapolate these measurements throughout the drainage network. Inside the "data" folder is a file titled "watershedname_D.csv", which is an example file for grain size distribution. The file contains just one column, "D", and each value is a grain size measurement. This file can be filled in with the users field data on grain size distributions, and modified so that the name reflects the location where the data was collected. The total load sediment transport equation used by SeRFE requires that grain size measurements be bulk measurements, not just the bed surface (i.e. the measurements should characterize the entire bed, including surface and subsurface layers). The file can also be replicated; there should be a unique file for each site where grain size information was collected. Additionally, the user should seek to spatially distribute locations of grain size sampling.

The “data” folder also contains a file called “watershedname_width.csv” which should be filled in with data on channel widths. These measurements can come from the field, or can be obtained from remotely sensed data like aerial imagery (e.g. in Google Earth), and ideally should come from a location near a stream gage, as width measurements are paired with discharge values in the table. The file contains three columns: “DA”, “Q”, and “w”. These signify drainage area (km^2), discharge (m^3/s), and channel width (m) measured at the specified discharge, and should be filled in for each site. Again, selected sites should be well distributed across the watershed if possible.

Hydrologic data: The process of filling in the “.csv” files related to basin hydrology for the model require that the user download available gage data within the basin and perform some basic hydrologic analysis. First, the two-year flood (Q_2) should be obtained for each gage that does not have significant hydrologic alteration (i.e. is not downstream of a dam). If there are no such gages in the basin, regional relationships can be used to estimate what a baseline Q_2 *would* be in the absence of alteration (https://streamstats.usgs.gov/ss/ is a good resource for obtaining this information). Then, for each gage where Q_2 has been estimated, fill in the “reference_Q2.csv” file. **Note**: the USGS uses ft^3/s, and SeRFE uses m^3/s, so Q_2 values must be converted.

Next fill in the “watershedname_hydrographs.cvs” file. In this file, each *row* is a unique gage. For each gage, fill in a name (column 1) and the segment ID (e.g. figure 2) where the gage is located along the drainage network being used for modeling (column 2). If the gage is not located on a network segment a no data value of -9999 can be entered. In the third column, “regulated”, enter a 0 if the gage is above dams and a 1 if it is downstream of and therefore influenced by a dam. Each subsequent value from left to right is the mean daily discharge, which again must be converted to cms if the data is being obtained from USGS gages.

At this point, all of the data needed to run SeRFE has been prepared.

3. **Pre-processing Tools**

The pre-processing tools are a set of tools that perform some basic geoprocessing to help prepare the drainage network for use with the subsequent sub-models and the main SeRFE model. The first two tools, “Watershed DEM” and “Drainage Area” offer alternative methods for performing some GIS tasks outside of GIS software. The third and fourth tools, “Network Topology” and “Network Attributes” are used to generate network attributes necessary for the rest of the modeling process.

The pre-processing tools are operated via a graphical user interface (GUI). To launch the GUI, open the SeRFE folder in your IDE, open the “preprocessing_gui.py” script and run the script (see figure 4). A dialog box with four tabs for the four different tools will open (figure 3). After running any of the four tools, the dialog box will close itself upon completion. To run another tool, simply run the script again.

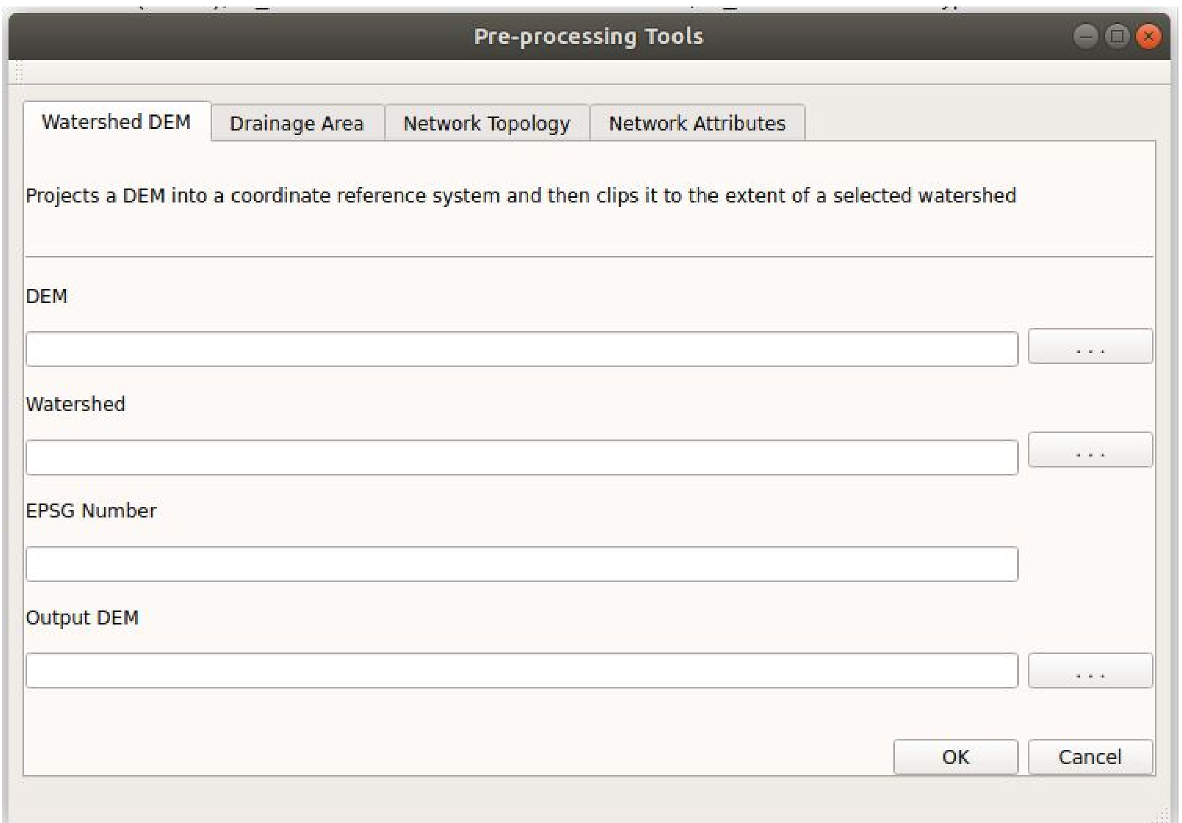

Figure 3. The GUI for the pre-processing tools.

Watershed DEM: The watershed DEM tool clips the mosaiced DEM to the boundaries of the watershed of interest, and projects it into a chosen CRS. This ensures that the data is projected and makes generating a drainage area raster from the DEM more computationally efficient.
<u>Inputs</u>:

- DEM: select the DEM mosaic from the “data” folder.

- Watershed: watershed boundary shapefiles are included with NHD downloads. Place the desired watershed boundary shapefile into the “data” folder and then select it here.
- EPSG Number: integer - the epsg number for the desired coordinate reference system (e.g. NAD 83 UTM 11 = 26911; for conversions see spatialreference.org).
- Output DEM: navigate to the “data” folder and store the DEM with a chosen name and extension (some_name.tif).

Drainage Area: The drainage area tool generates a raster from a DEM where the value of each cell represents the upstream contributing drainage area to that cell in km^2. The DEM *must be in a projected CRS*.

Inputs:

- DEM: navigate to the clipped DEM in the “data” folder.
- Drainage Area Raster Output: navigate to the “data” folder and store the drainage area raster with a chosen name and extension (some_name.tif).

Network Topology: The network topology tool adds 5 new attributes to the drainage network for SeRFE to determine topology (i.e. relative location of segments):

- ‘rid’: a reach identifier, unique to each segment.
- ‘rid_us’: the reach ID of the next segment upstream.
- ‘rid_us2’: the reach ID of the other upstream segment if the given reach is below a confluence, no data value of -9999 if there is only one upstream segment.
- ‘rid_ds’: the reach ID of the next segment downstream.
- ‘confluence’: 0 if the reach is *not* a confluence, 1 if the reach *is* a confluence.

Inputs:

- Drainage Network: select the prepared drainage network shapefile from the “data” folder.
- DEM: select the clipped DEM from the “data” folder.

Network Attributes: Adds 5 new attributes to the drainage network:

- ‘Drain_Area’: the contributing drainage area of the segment (km^2).
- ‘Slope_min (also mid and max)’: the reach averaged channel slope (unitless). The three different fields are updated individually as SeRFE runs to provide bounds upper and lower bounds on model outputs.
- ‘Sinuos’: Sinuosity of the segment (unitless). This attribute is used for bank erosion modeling.

Inputs:

- Drainage Network: select the drainage area shapefile from the “data” folder.
- DEM: select the DEM from the “data” folder.
- Drainage Area Raster: select the drainage area raster from the “data” folder (see Drainage Area Tool).
- EPSG Number: integer - the epsg number for the desired coordinate reference system (e.g. NAD 83 UTM 11 = 26911; for conversions see spatialreference.org).

4. Sub-models

Seven sub-models prepare the drainage network with the remainder of attributes necessary for running the main SeRFE model. These models are all run by running an associated script inside of the Python IDE. Each script name begins with “RUN” and scripts with this in the name are the only ones that need to be opened and modified at all to run the model. The scripts are set up so that all the user has to do is open them in an IDE, specify the paths to the required inputs, and click “run” in the IDE (figure 4).

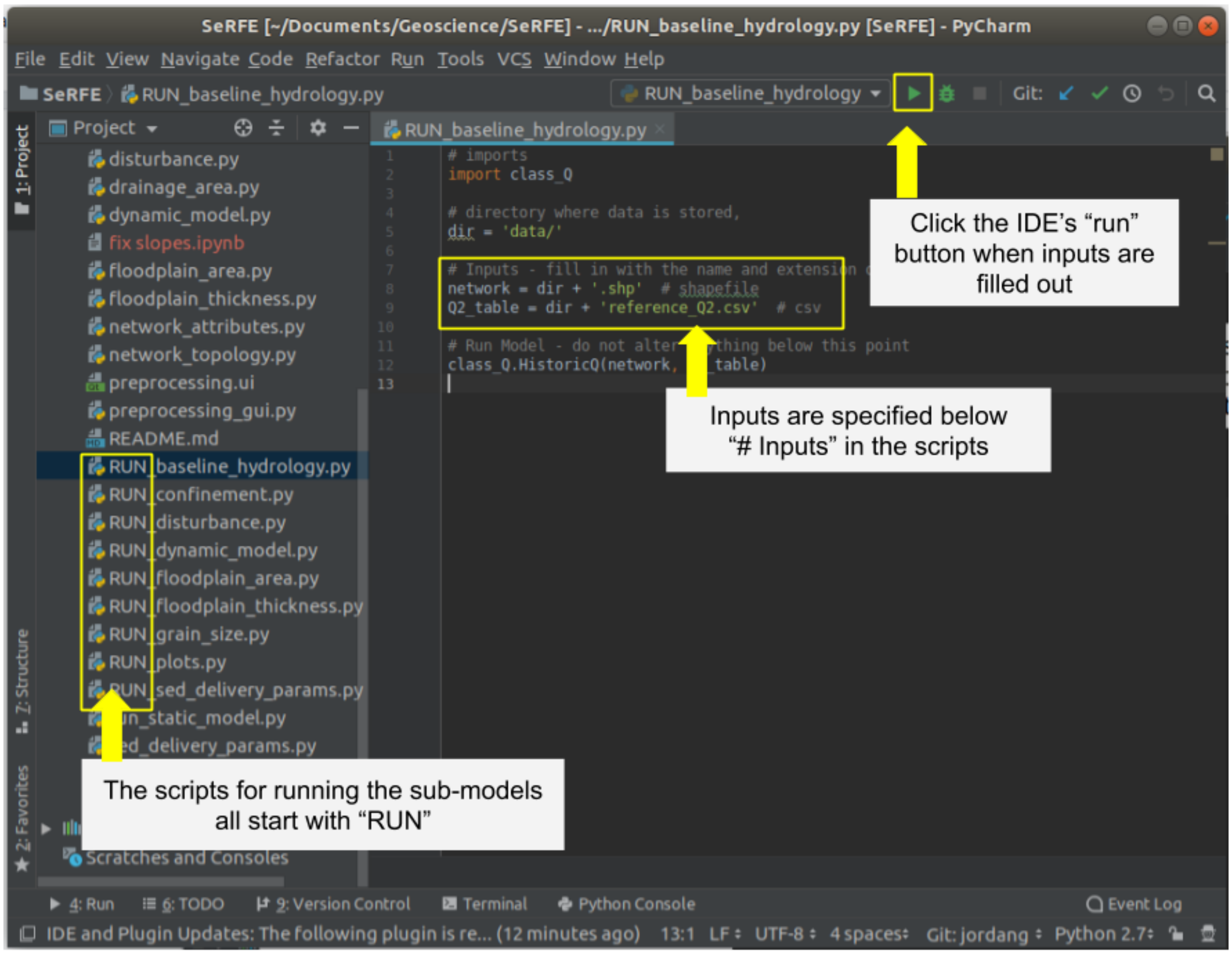

Figure 4. A sub-model script opened in the PyCharm IDE, highlighting where inputs are specified and the script run. The user should make sure that the desired script is what will run when the “run” button is clicked (it should be displayed in the drop-down menu next to the “run” button.

4.1. Baseline Hydrology (RUN_baseline_hydrology.py)

The baseline hydrology model uses the information contained in the “reference_Q2.csv” file to develop a drainage area-discharge relationship for the basin, which will be the basis for modeling discharge at each time step in the model. A regression is fitted to the data in the table and an output figure produced and saved in the “data” folder (figure 5). A ‘Q2’ attribute is added to the drainage network and filled with an estimated Q_2 value calculated for each segment.

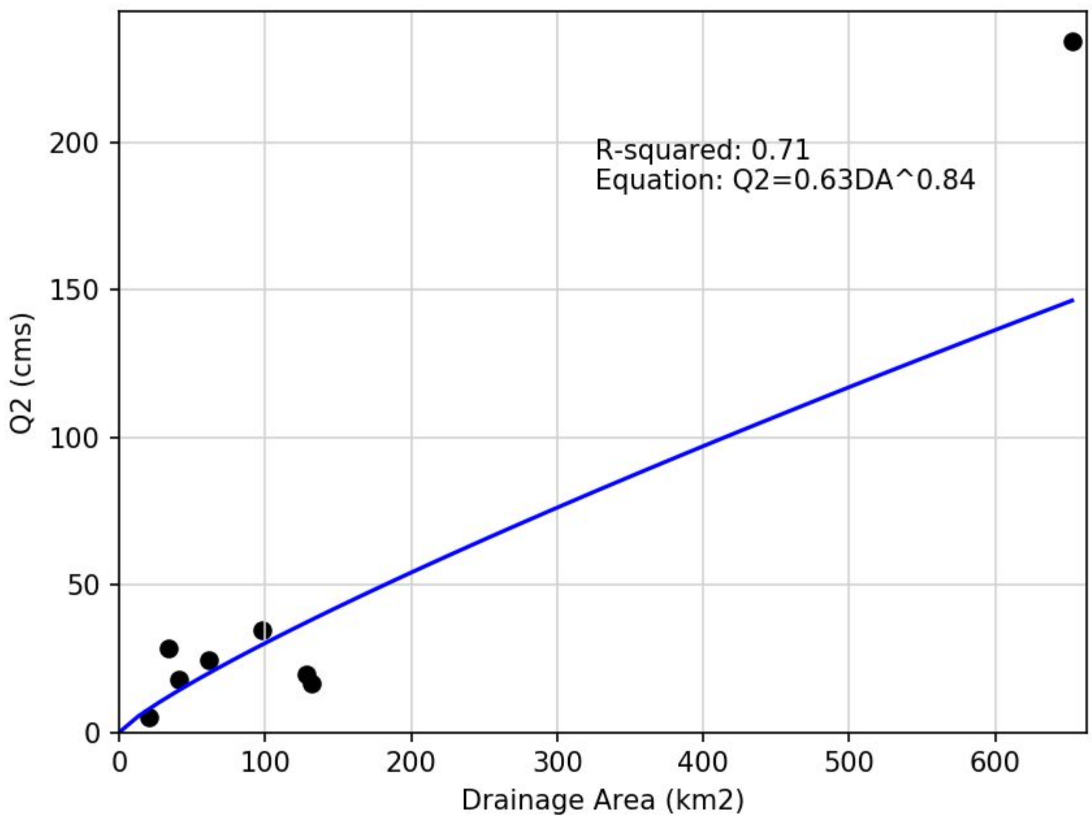

Figure 5. Figure produced for regression results for a Q_2 analysis using 8 gages.

<u>Inputs</u>:

- network: enter the name and extension (.shp) of the drainage network in the “data” folder.
- Q2_table: the name and extension of the table with the Q2 values for the basin’s gages. This is filled with “reference_Q2.csv” by default and should not need to be changed.

4.2. Grain Size (RUN_grain_size.py)

The grain size model uses an algorithm that extrapolates the grain size measurements from field sites to the rest of the drainage network. First, the unit stream power at which the bed is mobilized, based on the median grain size at each field site, is calculated using the equation:

$$\omega_c^* = \frac{\omega_c}{g(\rho_s - \rho)\sqrt{\frac{\rho_s - \rho}{\rho} g D^3}} \ . \ (1)$$

This equation derives a dimensionless form of stream power in the same way that Einstein (1948) derived a dimensionless transport rate. ω_c* is a dimensionless critical unit stream power, ω_c is critical unit stream power, g is acceleration due to gravity, ρ_s is the density of sediment ρ is the density of water and D is a median bed grain size. The authors that developed the equation found that ω_c* averaged 0.1 for the flume and natural river data that they used in their study (Parker et al., 2011).

At each site, the grain size measurements (D) can be plugged in, ω_c* assumed to be 0.1 and equation 1 solved for ω_c to calculate the critical unit stream power of the site. ω_c by definition can also be calculated as

$$\omega_c = \frac{\rho g Q_c S}{w} \ (2)$$

where Q_c is critical discharge, S is bed slope and w is channel width. This equation can be rearranged to solve for Q_c,

$$Q_c = \frac{w \omega_c}{\rho g S} \ (3).$$

By plugging the calculated value of ω_c into equation 3, the critical discharge for a segment is calculated using measured grain size and modeled width. Because these equations include width (*w*), the model uses the information in the ‘watershedname_width.csv’ table to generate a model (multiple linear regression) for predicting width based on discharge and drainage area, and attributes each segment with an estimated bankfull or critical width (used in these equations). The grain size model produces plots of grain size distributions for each site (figure 6). The model then bootstraps the data to come up with a range of median grain size estimates, and uses this range to compute a range of critical discharge values using the above process, and producing the plots in figure 7.

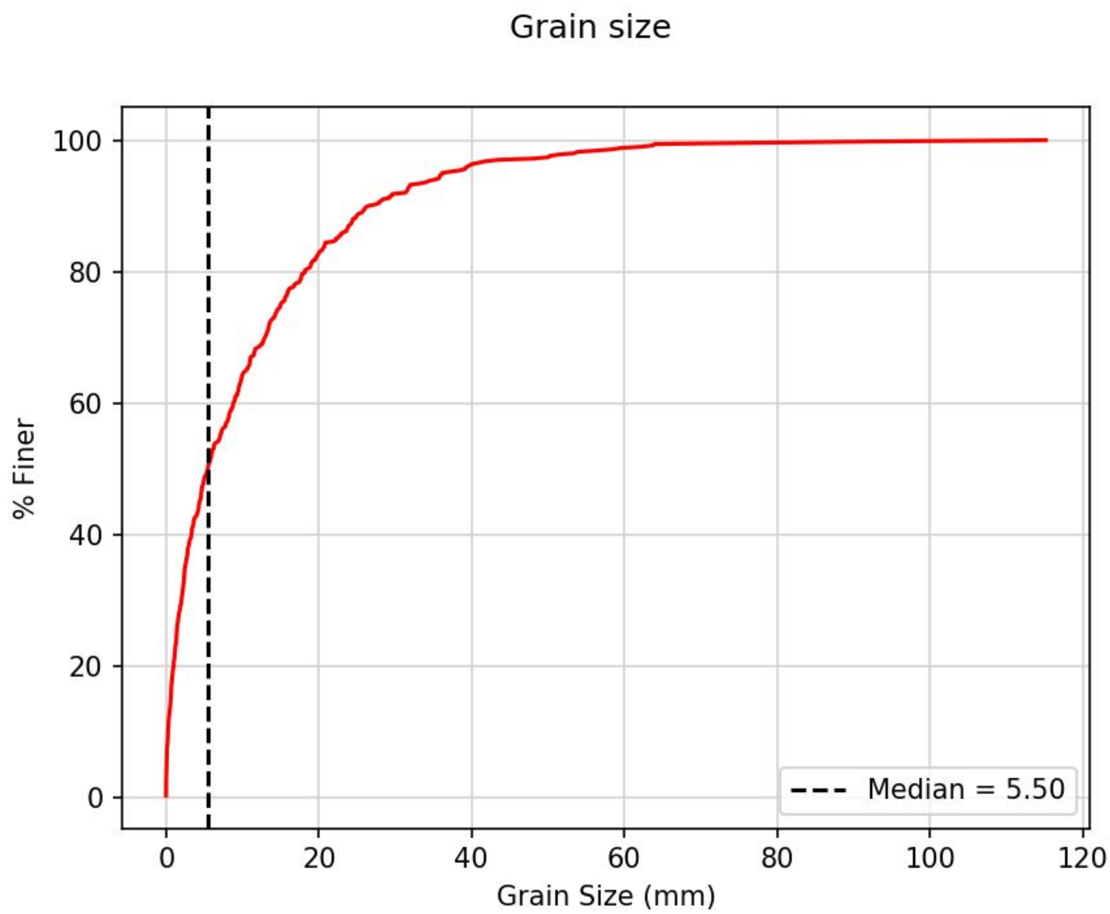

Figure 6. A grain size distribution plot produced by the grain size model for a specific site.

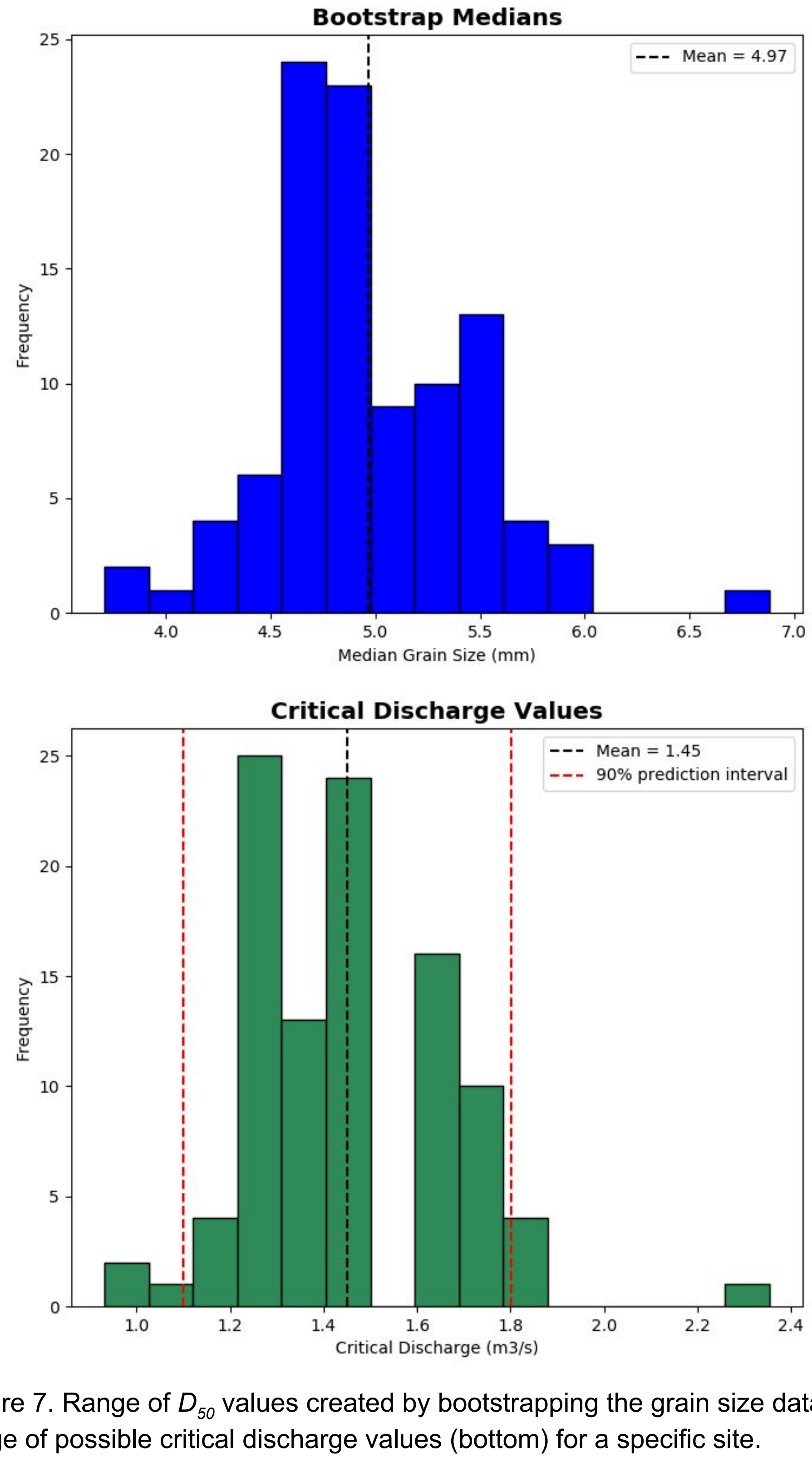

Figure 7. Range of D_{50} values created by bootstrapping the grain size data (top) and resulting range of possible critical discharge values (bottom) for a specific site.

For each site, ratios of the calculated minimum, mean and maximum critical discharge values (see figure 7) to the sites estimated Q_2 value is calculated. The average of these ratios for all sites is used to extrapolate three critical discharge values (minimum, mid, and maximum) to every segment of the drainage network. Equation 1 is then solved for *D* to derive

$$D_{pred} = \left(\frac{\left(\frac{\omega_c}{\omega_c^*}\right)^2}{(g(\rho_s - \rho))^2\left(\frac{\rho_s}{\rho} - 1\right)g}\right)^{\frac{1}{3}} \quad (4).$$

Using equation 4, three median grain sizes (in mm) are estimated for each segment of the drainage network. Three attributes "D_pred_(low, mid, high)" are added with these values for each segment.

Inputs:

- network: enter the name and extension (.shp) of the drainage network in the "data" folder.
- grain_size: a list of paths (name and extension .csv) for the grain size measurements for each individual site.
- reachids: a list of the segment IDs for each site where grain sizes were measured (again in the same order)
- width_table: the path (name and extension .csv) of the width table filled out in the data folder.

4.3. Floodplain Area (RUN_floodplain_area.py)

The floodplain area model calculates the area (m^2) of floodplain associated with each drainage network segment and adds an attribute, "fp_area" with this value.

Inputs:

- network: enter the name and extension (.shp) of the drainage network in the "data" folder.
- floodplain: enter the name and extension (.shp) of the floodplain/valley bottom polygon in the "data" folder.
- lg_buf: a maximum distance from the high drainage area (>250 km^2) portion of the network to search for floodplain features. Default 1500 m.
- med_buf: a maximum distance from the medium drainage area (25 - 250 km^2) portion of the network to search for floodplain features. Default 500 m.
- sm_buf: a maximum distance from the small (< 25 km^2) drainage area portion of the network to search for floodplain features. Default 50 m.

4.4. Confinement (RUN_confinement.py)

The confinement model automatically calculates valley confinement following Fryirs et al. (2016) for each segment using an algorithm modified from O'brien et al. (2019). Confinement is defined as the proportion of a drainage network segment that abuts a confining margin (edge of valley bottom), and is calculated by

$$C_V = \left(\frac{\sum_{DS}^{US} CL_{EB}@H_S}{CL_T}\right) \times 100 \quad (5)$$

where *US* and *DS* are the upstream and downstream extents respectively, C_V is the valley confining margin, $CL_{EB}@H_S$ is the length of channel along either bank that abuts a valley margin and CL_T is the total length of channel (figure 8). A value of 0 indicates that the segment is completely unconfined, and as a result free to migrate and interact with its floodplain. A value of 1 indicates that a segment is completely confined with no ability to adjust laterally. The model calculates this value for each segment and adds an attribute, "confine" with this value.

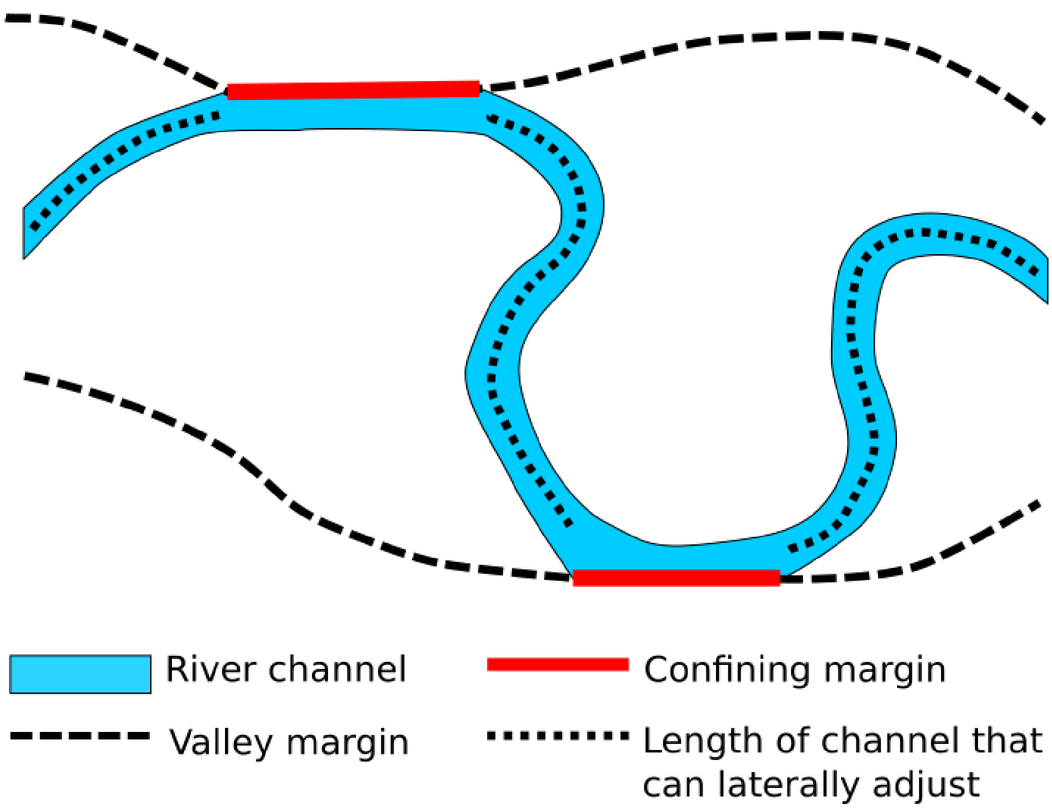

Figure 8. Conceptual diagram demonstrating how confinement is calculated.

<u>Inputs</u>:

- network: enter the name and extension (.shp) of the drainage network in the “data” folder.
- valley: enter the name and extension (.shp) of the floodplain/valley bottom polygon generated (which should also be stored in the “data” folder).

4.5. Floodplain Thickness (RUN_floodplain_thickness.py)

The floodplain thickness model estimates the depth of the floodplain at each drainage network segment with associated floodplain features. An average elevation within the low flow channel is first computed, then an average elevation of the floodplain. The difference between these two elevations is the estimated floodplain thickness, and three attributes “fp_thick_(min, mid, max)” are added to the network with this value. Similar to channel slope, this value adjusts as the model runs and sediment is recruited from or deposited to the floodplain, which is why there are three “trajectories” that get tracked to capture uncertainty.

Inputs:

- network: enter the name and extension (.shp) of the drainage network in the “data” folder.
- valley: enter the name and extension (.shp) of the floodplain/valley bottom polygon generated (which should also be stored in the “data” folder).
- dem: enter the name and extension of the clipped, projected DEM in the “data” folder.
- min_thickness: enter a value (m) for a minimum floodplain thickness.
- max_thickness: enter a value (m) for a maximum floodplain thickness.

4.6. Sediment Delivery Parameters (RUN_sed_delivery_params.py)

Hillslope sediment delivery is modeled as a function of erosion rates, which are selected from gamma distributions. Consequently the user should have familiarity with the parameters that govern gamma distributions prior to using this model. These two parameters, shape and scale govern the shape and possible values within the distribution. A minimum and maximum scale value are used to bound the upper and lower limits of erosion rates, and the shape parameter is used to determine the distribution of the range of possible values (more normal, or more right skewed). In the SeRFE model, erosion rates vary temporally by selecting a value from these distributions at each time step. They vary spatially by adjusting the gamma parameters between the maximum and minimum specified value based on local hillslope gradient, which is what this sub-model does (figure 9).

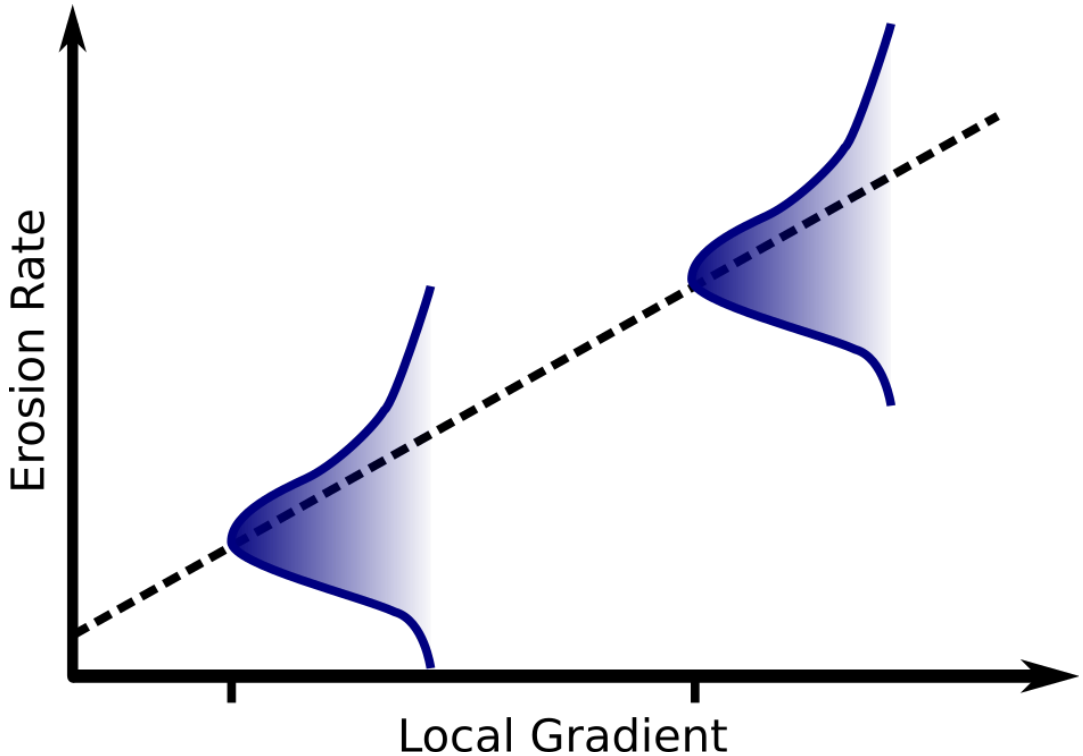

Figure 9. Conceptual figure showing how hillslope erosion rates are modeled in SeRFE.

The sediment delivery parameters sub-model attributes each segment of the network with a “g_shape” and “g_scale” parameter. As previously mentioned, the “g_scale parameter is a function of the local hillslope gradients, so slope must be calculated from the DEM. If the basin being modeled is small (e.g. HUC 12), this model can quickly automatically generate a slope raster. If it is larger, it is more efficient to generate a slope raster in a GIS and save the output to the “data” folder in the SeRFE model directory.

Inputs:

- dem: enter the name and extension of the clipped, projected DEM in the “data” folder.
- slope_out: if using the tool to generate a slope raster, specify a name and extension (.tif), and the file will be saved in the “data” folder. If using an externally generated slope raster, enter the name and extension that it was saved as in the “data” folder.
- network: enter the name and extension of the drainage area shapefile in the “data” folder.
- neighborhood: specify a distance (m) away from the network segments, within which the average hillslope gradient will be calculated. Default 500 m.
- g_shape: specify a gamma shape parameter.
- g_scale_min: specify a minimum gamma scale parameter.
- g_scale_max: specify a maximum gamma scale parameter.
- calc_slope: boolean - enter True if using the tool to generate a slope raster, False if not. Default False.

4.7. Disturbance (RUN_disturbance.py)

Disturbances are added to the network with a specified location and (in some cases) time. Disturbances can be in the form of a dam (reducing sediment supply and altering flow regime), or an event that increases sediment supply (e.g. wildfire or landslide). For dams, the segment ID of the dam location is specified, and the effective drainage area of all segments downstream is updated to reflect the impact of the dam. Hydrograph locations at the base of the dam allow the model to account for the dams effect on flow (If one does not exist, one can be synthesized using the best available knowledge of the flow regime). For increases in sediment supply, the segment ID is specified along with a start time, end time and new gamma erosion rate parameters.

Inputs:

- segid1: a list containing the segment IDs for reaches where dams are.
- new_da: boolean. Leave True to account for dams reducing effective drainage area of stream segments downstream.
- segid2: a list of segment IDs of reaches affected by a disturbance resulting in an *increase* in sediment supply.
- dist_start: disturbance starting time step. This can be a list of *either* a single value, if all segments listed in “segid2” have the same disturbance start time, *or* a list of the same length as “segid2” with a unique starting time step for each segment.
- dist_end: disturbance ending time step. This can be a list of *either* a single value, if all segments listed in “segid2” have the same disturbance end time, *or* a list of the same length as “segid2” with a unique end time step for each segment.
- new_denude: new gamma parameters for erosion simulation. This can either be a list of a single parameter set (e.g. [[3, 1]]) if all segments experience the same increase in erosion rate, or a list of the same length as “segid2” with a unique parameter set for each segment (e.g. [[3, 1][2.5, 1.5],[3.5,1]]).

5. **SeRFE Model (RUN_SeRFE.py)**

With all the sub-models run, the network is prepared for use with the main SeRFE model.

At each time step, for each segment the following occurs: 1) flow is calculated, and flow depth estimated using a specified Manning's roughness coefficient. (The user enters maximum and minimum values for Manning's *n* across the basin, and a linear model relating these maximum and minimum values to the maximum and minimum values of the grain size distribution for the basin is used to generate a Manning's *n* for each segment). 2) Sediment supplied from any adjacent upstream segment(s) and sediment contributed from neighboring hillslopes is accounted for, as well as in-channel storage remaining from previous time steps. 3) If the stream power for the calculated flow exceeds the segments critical stream power, any contributions from bank erosion are also accounted for. 4) Transport capacity is calculated using the Lammers and Bledsoe (2018) equation,

$$q_s = 0.0214(\omega - \omega_c)^{1.5} D^{-1} q^{-5/6} \quad (6)$$

where q_s is in ppm and q is width normalized discharge (*Q/w*). The balance between sediment transport capacity and supply in each segment is used to determine the fate of supplied sediment. If the transport capacity is less than supply, the amount that can be transported is exported from the segment, and the remainder is placed into storage. If the flow depth does not exceed the height of the floodplain, all storage occurs as in-channel storage. If the flow depth exceeds the floodplain height, the storage is partitioned between the channel and the floodplain based on the volume of water passing over each, and an estimated settling velocity (Church 2004). If channel storage occurs bed slope is updated, and if floodplain storage occurs the height of the floodplain is updated. If the transport capacity is greater than the sediment supply, all sediment supplied, including sediment recruited from the floodplain (corrected for overbank deposition that occurs), is transported downstream to the next segment. The volume recruited from the floodplain is then converted to a height and subtracted from the floodplain height. If the transport capacity is equal to the sediment supply, all sediment is moved through the segment with no changes.

This logic is repeated for each time step (length of flow record provided). At each time step, information is recorded and stored in an output table. The table has two index columns: time step and segment ID, which are used to store flow, sediment fluxes and sediment storage for each time step. Additionally a "capacity supply ratio" (CSR), defined as the calculated transport capacity divided by the calculated supply for a given time step is tracked (Soar and Thorne, 2001). CSR is simple to calculate and useful for showing variation in sediment balance through time. To characterize uncertainty in the outputs, the model performs this logic on three "trajectories" based on the prediction interval of bed grain size (a minimum, median and maximum trajectory). The network is automatically attributed with total sediment yield in tonnes

(‘Qs_out_min’, ‘Qs_out_mid’, ‘Qs_out_max’) and normalized sediment yield in tonnes/km^2 (‘Qs_nor_min’, ‘Qs_nor_mid’, ‘Qs_nor_max’). Change in storage in tonnes over a specified time interval, and the Capacity Supply Ration (CSR) integrated across a specified time interval and normalized by the length of the time interval, can also be added to the network output (see ‘.py’ description below).

Inputs:

- hydrograph: provide the name and extension (.csv) of the table in the data folder containing hydrographs for all gages in the basin.
- width_table: provide the name and extension (.csv) of the table in the data folder that contains measurements of width paired with discharge and drainage area.
- flow_exp: enter the value of the exponent in the flow equation (Q2). This can be found on the chart that is produced from the “baseline hydrology” tool (e.g. 0.84 in Figure 4).
- network: the name and extension (.shp) of the drainage network, which at this point should have been run with all of the sub-models in order to attribute it with all of the necessary attributes. Note: if the network does *not* contain all necessary attributes at this point, the code will throw an exception informing the user.
- mannings_min: enter a minimum Manning’s roughness coefficient for the basin (default 0.03).
- mannings_max: enter a maximum Manning’s roughness coefficient for the basin (default 0.05).
- bulk_density: enter a value to use for the bulk density of floodplain sediment deposits (default 1.2 tonnes/m^3).
- out_df: enter a name and extension (.csv) to store the table containing all output data.
- spinup: boolean - if True, no output will be saved. Instead the model will be run for the hydrographs provided and model feedbacks will allow the channel slopes and floodplain thicknesses to adjust. These adjustments will be saved to the network for subsequent model runs. If False, the model runs as normal and output table is saved. Several (~3-4) spinup runs are sometimes necessary to totally smooth out inputs and get sediment yield values.

6. **Analyzing outputs (RUN_output_analysis.py)**

A script is provided to help summarize model outputs and attribute the drainage network with additional attributes for visualization and analysis. Filling out different blocks (described below) allows you to produce different figures and adds different attributes to the drainage network.

Inputs:

- CSR: boolean - enter True if you wish to integrate the daily CSR value for each segment over a specified time interval. Three new attributes will be added to the network (‘CSR_min (mid, max);).

- CSR_start: int - specify the time step at which you want to start CSR integration. If CSR is True, this attribute must be filled out.
- CSR_end: int - specify the time step at which you want to end CSR integration. If CSR is True, this attribute must be filled out.
- Storage_plot: boolean - enter True if you wish to produce a plot of storage (floodplain and total) through time for a given drainage network segment (figure 10).
- Storage_segment: int - enter the segment ID. If storage_plot is True this value must be filled in.
- Store: boolean - enter True if you wish to calculate the change in storage for each segment over a specified time interval. Three new attributes will be added to the network ('d_Store_min (mid, max)').
- Store_start: int - specify the time step at which you want to start storage change calculation. If store is True, this attribute must be filled out.
- Store_end: int - specify the time step at which you want to stop storage change calculation. If store is True, this attribute must be filled out.
- Date_fig: boolean - enter True if you wish to produce a figure of a specified attribute at a specified time step, and add an attribute for that time step to the network.
- Time_step: int - enter the time step
- Attribute: string - enter the attribute name from the SeRFE attribute table you wish to produce the figure of.
- CSR_series: boolean - enter True if you wish to produce a plot of calculated CSR through time for a specified drainage network segment (figure 11).
- CSR_segment: int - enter the drainage networks segment ID. if CSR_series is True, this value must be filled out.

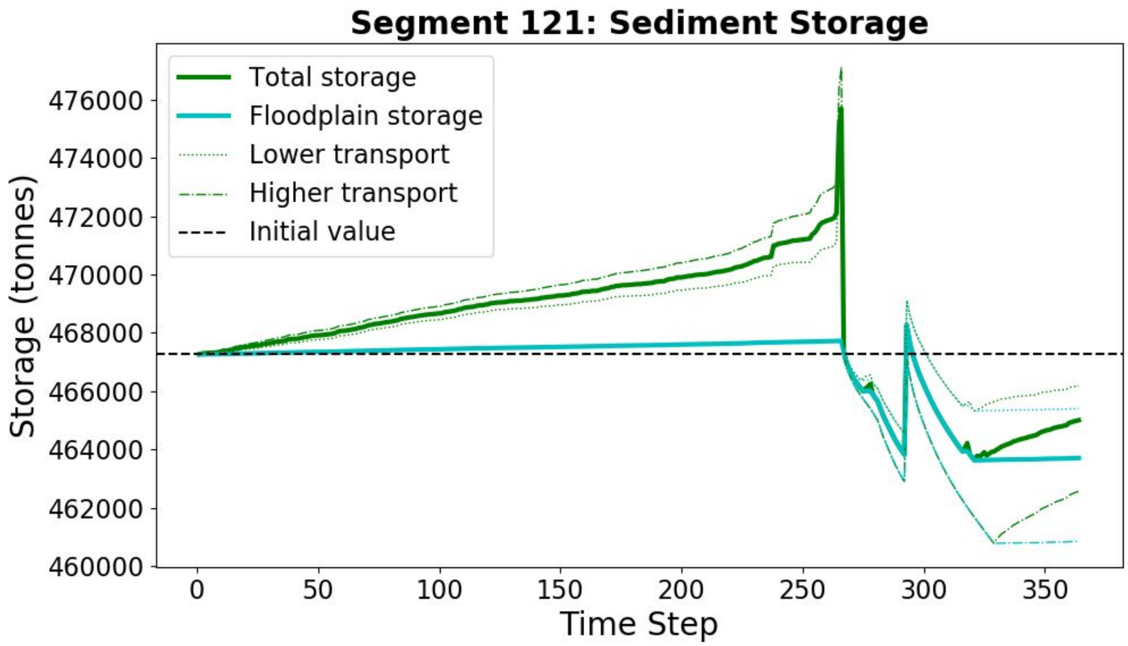

Figure 10. An example of a storage plot for a given drainage network segment.

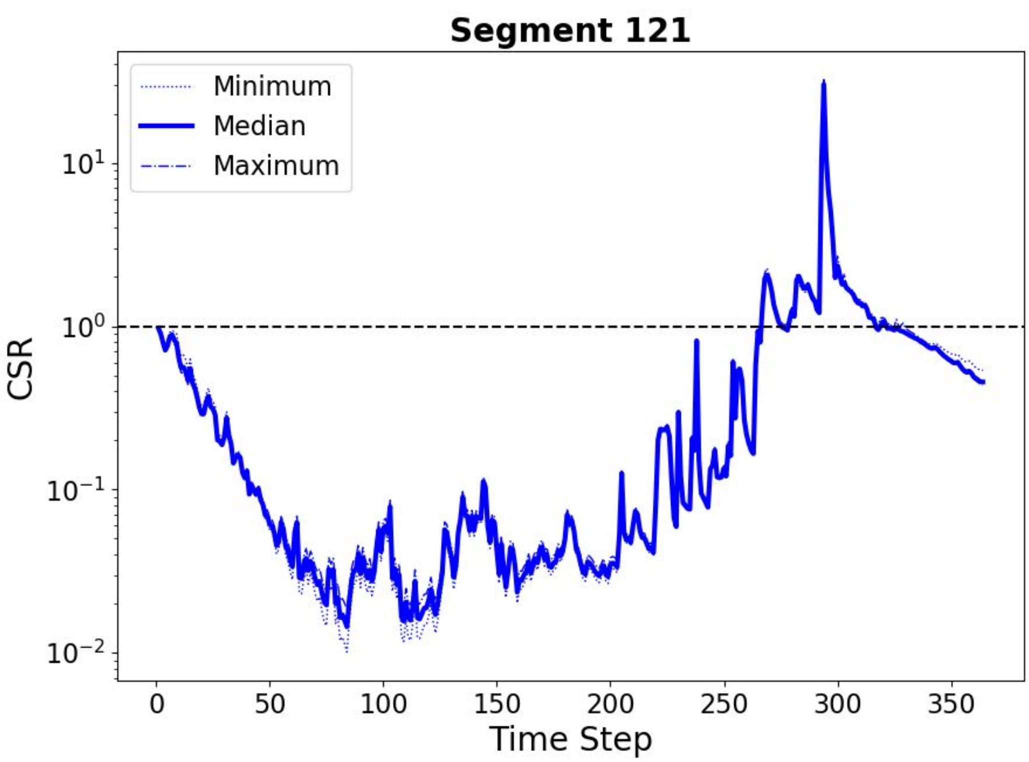

Figure 11. An example of a CSR plot for a given drainage network segment.

Bank Erosion Calibration

Bank erosion in SeRFE is modeled using a stream power version of a common erosion model,

$$\varepsilon = k(\omega - \omega_{c\ bank})^{a} \quad (6)$$

Where ε is erosion rate (m/s), *k* is an erodibility coefficient (m^3/W), ω is unit stream power of the flow and $\omega_{c\ bank}$ is the unit stream power required to initiate bank erosion. This is often used as a linear model, where *a*=1, but other times as a non-linear model (Wilson, 1993a, 1993b; Khanal, 2016) where *a*=0.5. For our application we used the non-linear version.

For our initial application of SeRFE, we used the Santa Clara River basin in Southern California. As such, the bank erosion equation was calibrated using measurements in this watershed, and will likely need to be modified/updated for use in other basins. We used five study reaches where LiDAR data was available from 2015. At these sites, in summer 2018, we performed photogrammetric surveys with a UAV, from which updated, high-resolution topographic datasets were generated. At each site, we delineated the active channel in GIS for both the 2015 and 2018 datasets. We delineated the area eroded between the two surveys, and then divided this area by the channel length to get an average distance of bank erosion for each site.

We then compiled flow information for each of the sites (mean daily flow (*Q*) for the year encompassing the 2017 flood). Additionally, for each site, we generated a model relating flow width to discharge by coupling measurements with aerial imagery in Google Earth. We used this model to estimate flow width (*w*) for each day. Slope (*S*) was retrieved from the DEM, and we calculated unit stream power for each day as

$$\omega = \frac{\rho g Q S}{w} \ . \ (7)$$

For each site then, a table was produced with headings:

Date	Q cms	SP	w_est	unit_SP

Where SP is stream power, w_est is the estimated width for the given flow and unit_SP is the unit stream power (ω), with an entry (row) for each date. Each site also had an associated median grain size and channel sinuosity.

We used the grain size measurements for each site to estimate critical stream power for the initiation of bank erosion using the equation:

$$\omega_{c\ bank} = 1.2(\omega_c^* (\rho_s - \rho)\sqrt{\frac{\rho_s - \rho}{\rho} g D^3}) \quad (8)$$

where the coefficient 1.2 relates $\omega_{c\ bank}$ to $\omega_{c\ bed}$ for our study basin, and the remainder of the equation is the Parker et al. (2011) method for calculating ω_c. The coefficient 1.2 is an assumption that results in low safety factors (a measure of bank strength compared to applied stress) at bankfull flows (Abernethy and Rutherfurd, 2000). In areas with higher clay content and denser vegetation, values of $\omega_{c\ bank} >> \omega_{c\ bed}$ would be expected, which would result in higher safety factors at bankfull flows.
This coefficient can be manually changed when running the model in other basins by opening the **'dynamic_model.py'** scripts and changing the value from 1.2 to the desired value in lines **304**, **311**, and **318**.

With $\omega_{c\ bank}$ calculated for each study site, equation 6 is used to estimate daily bank erosion by estimating *k* and multiplying the results of the equation by 86,400 (the number of seconds in a day). To find *k* for each site, an optimization algorithm was then used to vary *k* until the error was minimized between the sum of calculated bank erosion over all dates and the measured bank erosion from the DEMs was minimized. Finally, because *k* and $\omega_{c\ bank}$ both related to soil properties and are therefore related to each other, we fit a linear regression to predict *k* based on $\omega_{c\ bank}$. Initial model fit was relatively poor ($r^2 = 0.27$), but by adding sinuosity (which is added to the network in the preprocessing tools), a proxy for radius of curvature, model fit was substantially improved ($r^2 = 0.94$ based on our 5 data points). The resulting model to predict *k* is hard coded into SeRFE in the 'dynamic_model.py' file at lines 291-311 and is:

$$k = -6.04e^{-7} + 5.52e^{-8}\omega_{c\ bank} + 5.95e^{-7}\zeta \quad (9)$$

Where ζ is segment sinuosity.

If the user performs this routine on data for a different basin to update bank erosion behavior, the equation for calculating *k* should be updated in lines **305**, **312**, and **319** of the **'dynamic_model.py'** script.

In summary, for each of the five sites in the tables mentioned above, the values for $\omega_{c\ bank}$, calculated using grain size measurements, and measurements of bank erosion were used to calibrate the erosion coefficient, *k*. A model was then generated to predict *k* based on both $\omega_{c\ bank}$ and sinuosity, which was added to SeRFE, enabling modeling of bank erosion for each segment at each time step in the model. The tables, additional site information, and script used for this calibration can be found here:
https://drive.google.com/open?id=1PMu_dEDZKFSO6oMjeoBdMJnzudJmss-e

References

Abernethy, Bruce, and Ian D. Rutherfurd. "The effect of riparian tree roots on the mass-stability of riverbanks." Earth Surface Processes and Landforms: The Journal of the British Geomorphological Research Group 25.9 (2000): 921-937.

Einstein, H. A., “Formulas for the transportation of bed load.” Transactions of ASCE (1941), 561-597.

Ferguson, R. I., and M. Church. "A simple universal equation for grain settling velocity." Journal of sedimentary Research 74.6 (2004): 933-937.

Fryirs, Kirstie A., Joseph M. Wheaton, and Gary J. Brierley. "An approach for measuring confinement and assessing the influence of valley setting on river forms and processes." Earth Surface Processes and Landforms 41.5 (2016): 701-710.

Gilbert, Jordan T., William W. Macfarlane, and Joseph M. Wheaton. "The Valley Bottom Extraction Tool (V-BET): A GIS tool for delineating valley bottoms across entire drainage networks." Computers & geosciences 97 (2016): 1-14.

Khanal, A., et al. "Comparison of linear and nonlinear models for cohesive sediment detachment: Rill erosion, hole erosion test, and streambank erosion studies." Journal of Hydraulic Engineering 142.9 (2016): 04016026.

Lammers, Roderick W., and Brian P. Bledsoe. "Parsimonious sediment transport equations based on Bagnold's stream power approach." Earth Surface Processes and Landforms 43.1 (2018): 242-258.

O'Brien, Gary R., et al. "Mapping valley bottom confinement at the network scale." Earth Surface Processes and Landforms (2019).

Parker, Chris, Nicholas J. Clifford, and Colin R. Thorne. "Understanding the influence of slope on the threshold of coarse grain motion: Revisiting critical stream power." Geomorphology 126.1-2 (2011): 51-65.

Soar, P. J. and C. R. Thorne, “Channel restoration design for meandering rivers.” Engineer Research and Development Center, Vicksburg MS Coastal and Hydraulics Lab (2001).

Wilson, Bruce N. "Development of a fundamentally based detachment model." Transactions of the ASAE 36.4 (1993a): 1105-1114.

Wilson, Bruce N. "Evaluation of a fundamentally based detachment model." Transactions of the ASAE 36.4 (1993b): 1115-1122.