Primer for Rosgen Stream Classification Course Module

Spring 2021

Prepared by Zach Hilgendorf (MSci)
PhD Student, Geography
School of Geographical Sciences and Urban Planning
Arizona State University

Purpose

The purpose of this primer is to establish the goals, methods, data, and software required to complete the module for reproducibility in the Rosgen Stream Classification method using geographic information software. A series of short lectures and handouts have been developed to provide ample background information and the skills to necessary to complete the assignment. Two scientific articles have also been provided as required reading.

Background Information

Classification Methods:

To set the stage, classification schemes, whether in geography, economics, biology, etc., are designed to try and "bestow order onto chaos." If we can generalize a system, we can make certain assumptions about how a similar system may "behave" somewhere else. In geography, we adhere to Tobler's First Law, which states that things closer to one another are more alike than things that are farther apart. However, we need to be able to compare differences and similarities between streams across the world. If we can classify a stream, based on a number of measured parameters, and we find similar streams, in similar systems, in different parts of the county or world, then we can start compare those streams. If we are trying to modify the stream, for example, we may need to have an idea of whether the modification we are making will work for their designed goal. We use classification methods, like the Rosgen Stream Classification system, to that end. One of the primary uses of the Rosgen system is to inform on the "type" of stream for planned restoration projects. Many local, state, and federal agencies employ this method and train their scientists to use it. As such, there is an important economic aspect at stake. Communities or other agencies have to allocate funds to pay for restoration efforts and depend on these classification schemes to accurately characterize what type of restoration will best fit their problem.

Two scientific articles have been provided as required reading for this module. The first, Rosgen (1994), is one of the original papers that defines and describes the steps necessary to complete the classification of a stream, based on a series of measured in situ parameters. This paper should serve as the baseline of knowledge for you to complete the module. The second, Kasprak et al. (2016), examines a number of different stream classification schema, including Rosgen (1994), to assess the parameters that each uses to classify a stream. Together, these articles offer a succinct view of how classification schemes in fluvial systems handle a wide variety of parameters, how the inclusion or exclusion of variables differs, and why it is useful to have ways in which to classify streams.

Physical Geography:

Streams and rivers dissect the world around us, fill reservoirs, water crops, provide vital ecosystems for plants and animals, and have seen widespread degradation over the last 150 years. As we have mentioned, the Rosgen system has seen extensive use in the field of stream restoration. The topic of fluvial geomorphology and hydrology is crucial to understanding the parameters we are collecting for the Rosgen system and what they mean. A series of five short (5-10 minute) lectures have been recorded and uploaded to YouTube to help teach you some of the essential concepts you will need to know to complete this module. Each of these videos takes a look at a single topic (channel form, sinuosity, longitudinal profiles, grains+transport, ratios) and breaks them down to an introductory level.

Channel Form: https://youtu.be/OUT3emNhVwg
Sinuosity: https://youtu.be/rVs4MSw1e5s
Longitudinal Profiles: https://youtu.be/SK2BFg2PmlQ
Grains and the Initiation of Motion: https://youtu.be/61GNZwwuI4U
Ratios in Fluvial Geomorphology: https://youtu.be/ex7E5GHPp6Q

Software Skills:
In an effort to make this module widely accessible, we have chosen to use the open source GIS platform GRASS as the preferred graphical interface. GRASS is an incredibly powerful and customizable GIS platform, with a devoted and active user base, widespread integration with other software platforms (e.g., QGIS, RStudio), and free and open source software and code. The handouts in this lab are designed to teach you how to install and set up GRASS, import and manipulate raster and vector datasets, and process data with base and imported packages. These handouts should be approached in the following order:

1) Starting In GRASS
2) Digitizing in GRASS
3) Transects and Profiles
4) Classifying A Stream

Later on, you will be asked to do some very simple coding in RStudio, which is a user-friendly “wrapper” for the R language, developed by the Comprehensive R Analysis Network (CRAN). This code has been heavily commented and set up to allow you to understand what the code is doing and why you are inputting the variables you are in the places you are putting them. To install RStudio, you first need to install R. We will be working out of an RNotebook, which allows us to run “chunks” of code at a time and split up the goals of each chunk in a sensible way. All download sites have been provided below.

GRASS GIS: https://grass.osgeo.org/download/
CRAN: https://cran.r-project.org/
RStudio: https://rstudio.com/products/rstudio/download/#download

1 Getting Started in GRASS GIS

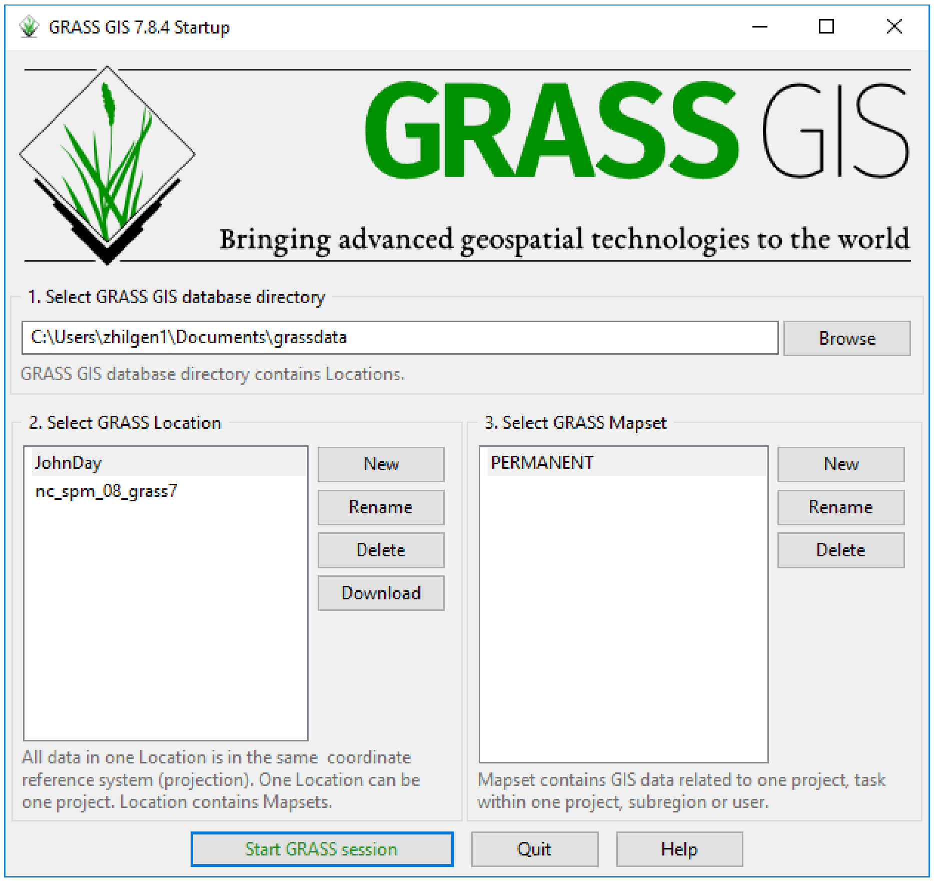
Purpose: The purpose of this handout is to teach you how to create a new project in the GIS software GRASS.

1.1 Creating a GRASS Project

When you first open GRASS GIS, you will see a black command prompt screen and a GUI that lets you set up and create new **Locations**.

GRASS NTK #1: GRASS has its own topological data structure and database system for managing geographic data. Therefore, working in GRASS requires:

- creating a **database directory** in which all GRASS data will be stored
- creating a **Location** establishing the geographic framework for a project or set of projects, including the coordinate reference system to be used and default extents and spatial resolutions
- creating a **Mapset** within a Location. We can use the default PERMANENT mapset, but you can organize different sets of geographic data layers for different projects within the same Location geographic frame by using mapsets.



1. Select GRASS GIS database directory

C:\Users\zhilgen1\Documents\grassdata

GRASS GIS database directory contains Locations.

2. Select GRASS Location

JohnDay
nc_spm_08_grass7

All data in one Location is in the same coordinate reference system (projection). One Location can be one project. Location contains Mapsets.

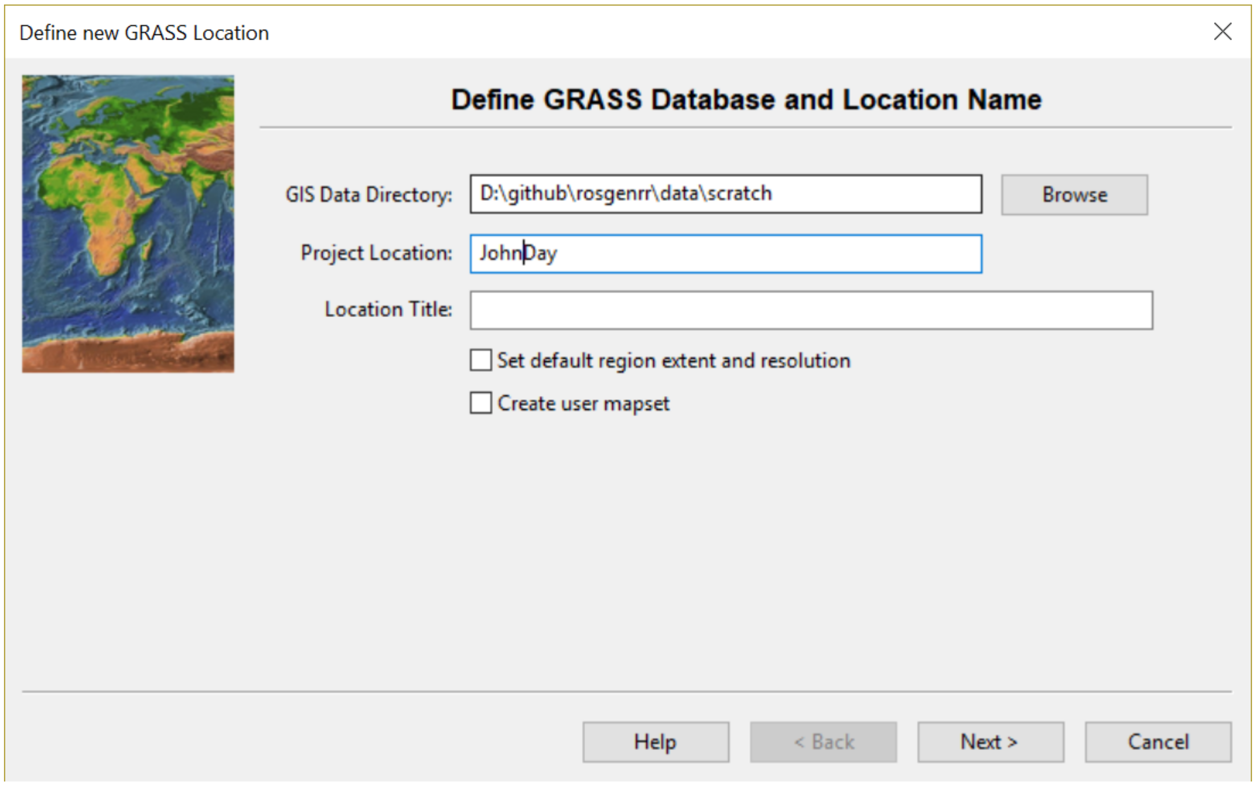
3. Select GRASS Mapset

PERMANENT

Mapset contains GIS data related to one project, task within one project, subregion or user.

Figure 1. The GRASS GIS startup window, where **directories**, **locations**, and **mapsets** can be created.

We will start by setting up a new **Location**. Click on New next to the "Select GRASS Location" box. When you do, a new window will open (see Figure 2). This allows you to create the default directory and name for the location. Set your default directory to the **scratch folder** in your **rosgenrr repository**, and give the project location a name. This could be the projection you are using (i.e. NAD 1983 (2011) UTM Zone 11N) or it could be more specific to the project. In this example, I called the location "**JohnDay**" in reference to the John Day River of Oregon, where our data is from.


Define new GRASS Location
Define GRASS Database and Location Name

GIS Data Directory: D:\github\rosgenrr\data\scratch

Project Location: JohnDay

Location Title:

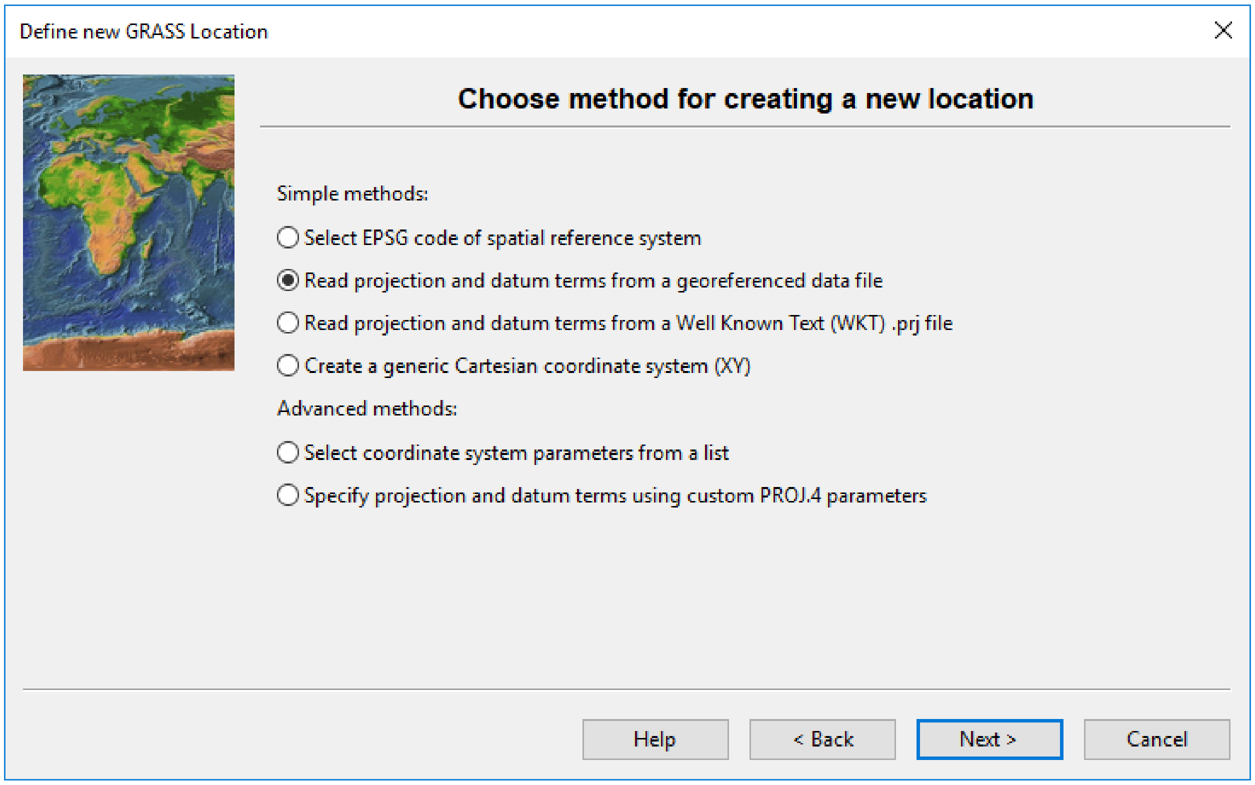
☐ Set default region extent and resolution

☐ Create user mapset

Browse
Help
< Back
Next >
Cancel

Figure 2. Window for setting up a new Location in GRASS.

Next, you will need to determine which **method** you want to use for **creating the new location**. GRASS has a lot of great ways to do this, from reading the European Petroleum Survey Group (EPSG) code, to reading the data directly from a file. We will employ the latter option for our project! Select the "Read projection and datum from a georeferenced data file" bubble and hit Next (Figure 3). In the next window, browse until you find the dataset you want to import the projection from. We will choose the CHaMP_Data_MFJD.prj file, as it is the projection for the CHaMP dataset you will be working with. Click Next after you have brought that file into the input box and you will be brought to a window that shows you the projection data that is being pulled from the file (Figure 4). Ensure that it is what you know the dataset to be. In this case, we are working in NAD 1983 UTM Zone 11 North. You will see that, in Figure 4, that "proj = utm" and "zone = 11," confirming that we imported the projection from the file.


Define new GRASS Location

Choose method for creating a new location

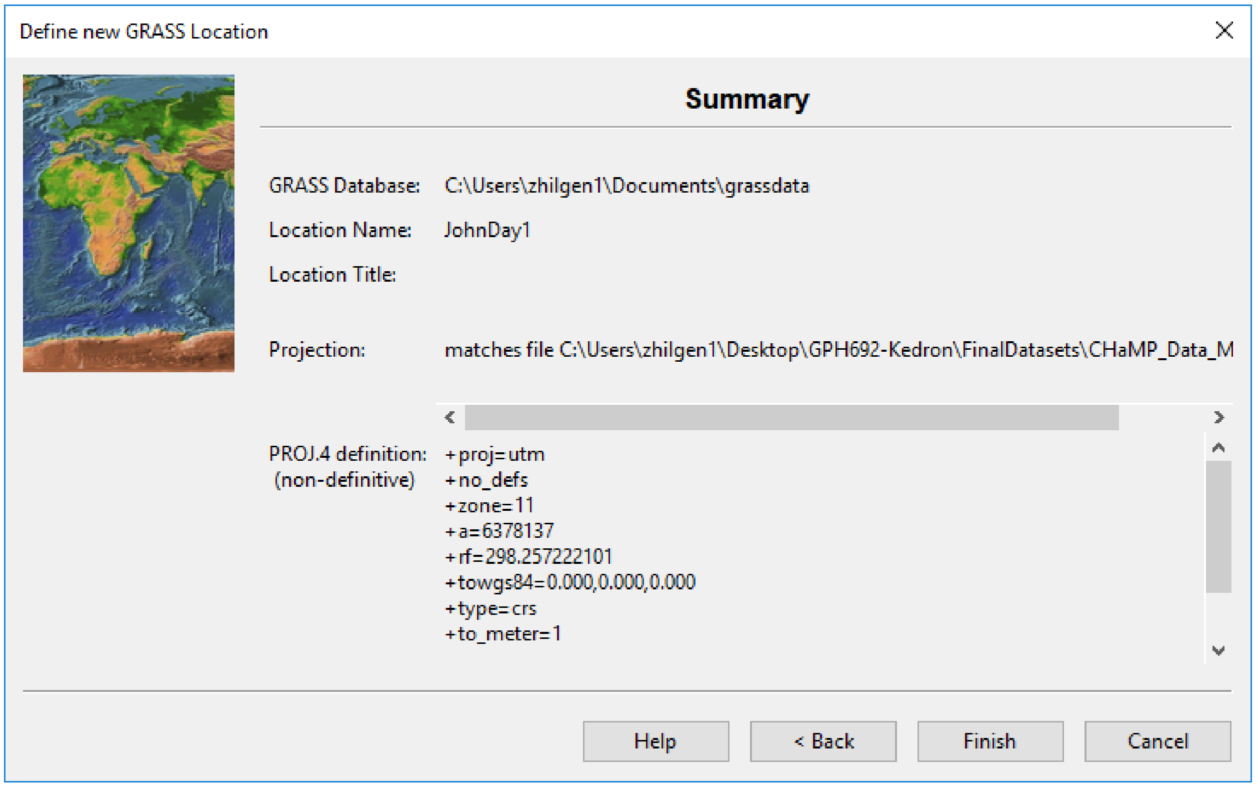
Simple methods:

( ) Select EPSG code of spatial reference system
[✓ Read projection and datum terms from a georeferenced data file]
( ) Read projection and datum terms from a Well Known Text (WKT) .prj file
( ) Create a generic Cartesian coordinate system (XY)

Advanced methods:

( ) Select coordinate system parameters from a list
( ) Specify projection and datum terms using custom PROJ.4 parameters

Figure 3. The window to choose how a location will acquire its spatial reference.


Define new GRASS Location

Summary

GRASS Database: C:\Users\zhilgen1\Documents\grassdata

Location Name: JohnDay1

Location Title:

Projection: matches file C:\Users\zhilgen1\Desktop\GPH692-Kedron\FinalDatasets\CHaMP_Data_M

PROJ.4 definition: (non-definitive)

```
+proj=utm
+no_defs
+zone=11
+a=6378137
+rf=298.257222101
+towgs84=0.000,0.000,0.000
+type=crs
+to_meter=1
```

Figure 4. The summary window that shows the projection that is being pulled from the chosen file.

Once you have confirmed that the projection is correct, select “Finish” and the location will generate. You will get a popup window that asks if you want to import the dataset you used as a reference into the Mapset. Either choice is fine, but this is a nice way to get a jump on importing your data into GRASS!

For this project, we can use the default PERMANENT mapset. Select “**Start GRASS Session**” to launch the program!

GRASS NTK #2: GRASS uses the many windows approach.

- The **Layer Manager** is the workhorse:
    - **Layers** tab is the equivalent to the Layers panel in QGIS, controlling display and order of layers in the map display
    - **Console** tab is the equivalent to the Log & History in QGIS, and also contains a command prompt where you type and run commands one algorithm at a time.
    - **Modules** tab is the equivalent of the Processing Toolbox: search and run algorithms interactively from here.
        - When you launch algorithms this way, a window opens and stays open even after you run the algorithm, making it easy to re-run the algorithm with the same settings and cluttering your desktop with excessive windows.
    - **Data** tab is the equivalent of browser, showing all of the spatial data in your current GRASS location.
- The **Map Display** window is where you view and interactively select, query, or digitize spatial data.

GRASS NTK #3: A **workspace** in GRASS is the equivalent of a Project in QGIS.

- Save your configuration of a map layout and layers with **File → Workspace → Save**
- Load your configuration of a map layout and layers with **File → Workspace → Load**

GRASS NTK #4: GRASS is *extremely particular* in managing its own spatial data structures. Therefore, you need to process all of your spatial data with an import procedure.

1.2 Loading and Visualizing Data

Go up to the **File** menu at the top of the Layer Manager window. Expand the “**Import raster data**” and select the top option “**Simplified raster import with reprojection**” (Figure 5). This can also be found by clicking on the Modules tab at the bottom of the Layer Manager window and typing in **“r.import”**, which is the abbreviated tool name.

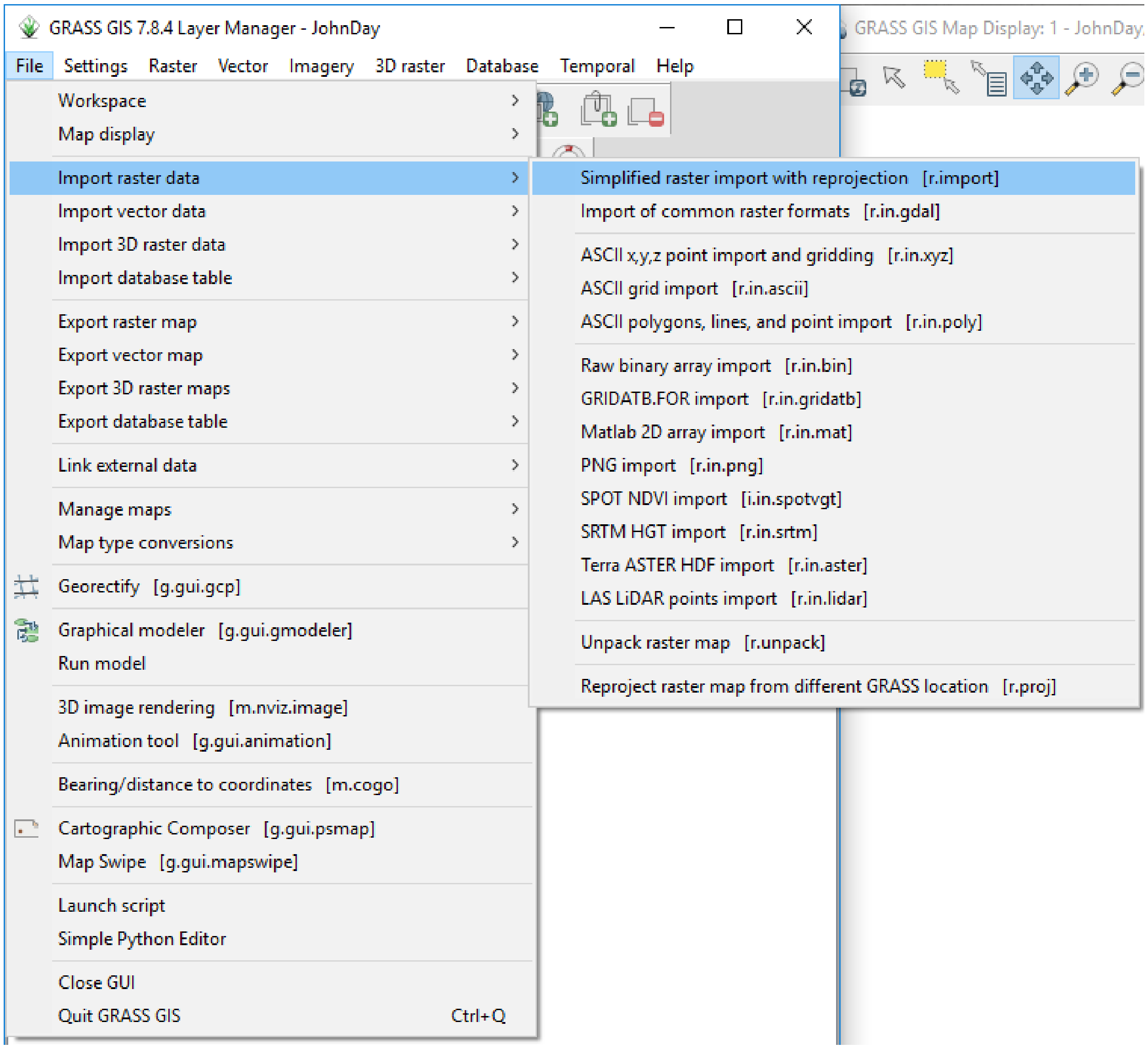
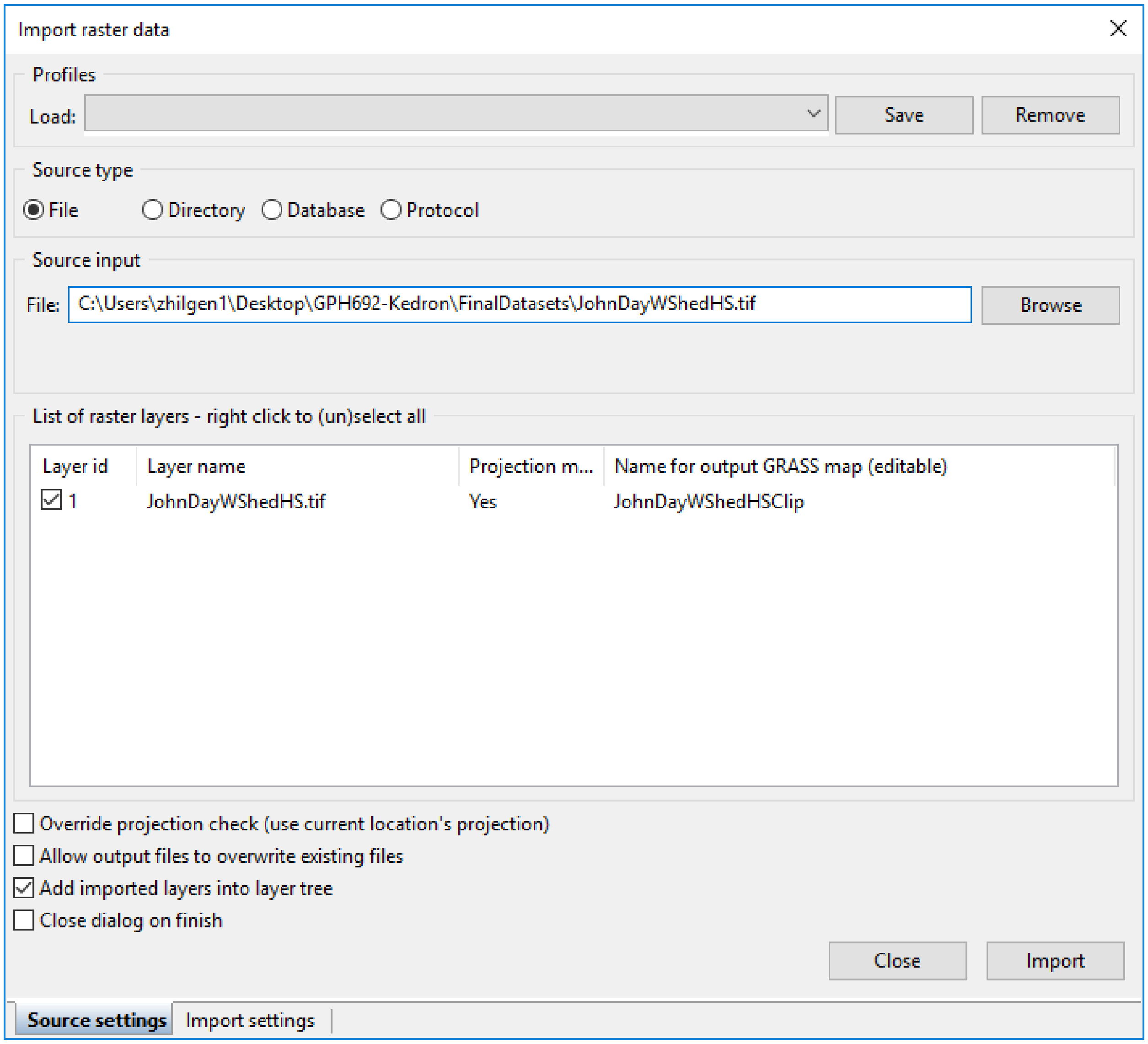

GRASS GIS 7.8.4 Layer Manager - JohnDay
GRASS GIS Map Display: 1 - JohnDay
File
Settings
Raster
Vector
Imagery
3D raster
Database
Temporal
Help
Workspace
Map display
Import raster data
Import vector data
Import 3D raster data
Import database table
Export raster map
Export vector map
Export 3D raster maps
Export database table
Link external data
Manage maps
Map type conversions
Georectify [g.gui.gcp]
Graphical modeler [g.gui.gmodeler]
Run model
3D image rendering [m.nviz.image]
Animation tool [g.gui.animation]
Bearing/distance to coordinates [m.cogo]
Cartographic Composer [g.gui.psmap]
Map Swipe [g.gui.mapswipe]
Launch script
Simple Python Editor
Close GUI
Quit GRASS GIS Ctrl+Q
Simplified raster import with reprojection [r.import]
Import of common raster formats [r.in.gdal]
ASCII x,y,z point import and gridding [r.in.xyz]
ASCII grid import [r.in.ascii]
ASCII polygons, lines, and point import [r.in.poly]
Raw binary array import [r.in.bin]
GRIDATB.FOR import [r.in.gridatb]
Matlab 2D array import [r.in.mat]
PNG import [r.in.png]
SPOT NDVI import [i.in.spotvgt]
SRTM HGT import [r.in.srtm]
Terra ASTER HDF import [r.in.aster]
LAS LiDAR points import [r.in.lidar]
Unpack raster map [r.unpack]
Reproject raster map from different GRASS location [r.proj]

Figure 5. Location of the raster import tools. The vector import tools are directly below.

This will open a new window, where you can Browse to select the raster dataset you want to bring in (Figure 6). Bring in the **JohnDayWshed** and **JohnDayWShedHS** files. These are the LiDAR digital elevation model and associated hillshade, respectively. Make the that the projection matches (it will say in the list of raster layers) and select an output name. Once you have finished setting things up, click import and let the tool work. These are larger files, so it will take a few minutes to import everything.


Import raster data

Profiles

Load: [ ] Save Remove

Source type

[✓ File] Directory Database Protocol

Source input

File: C:\Users\zhilgen1\Desktop\GPH692-Kedron\FinalDatasets\JohnDayWShedHS.tif

List of raster layers - right click to (un)select all

Layer id	Layer name	Projection m...	Name for output GRASS map (editable)
☑ 1	JohnDayWShedHS.tif	Yes	JohnDayWShedHSClip

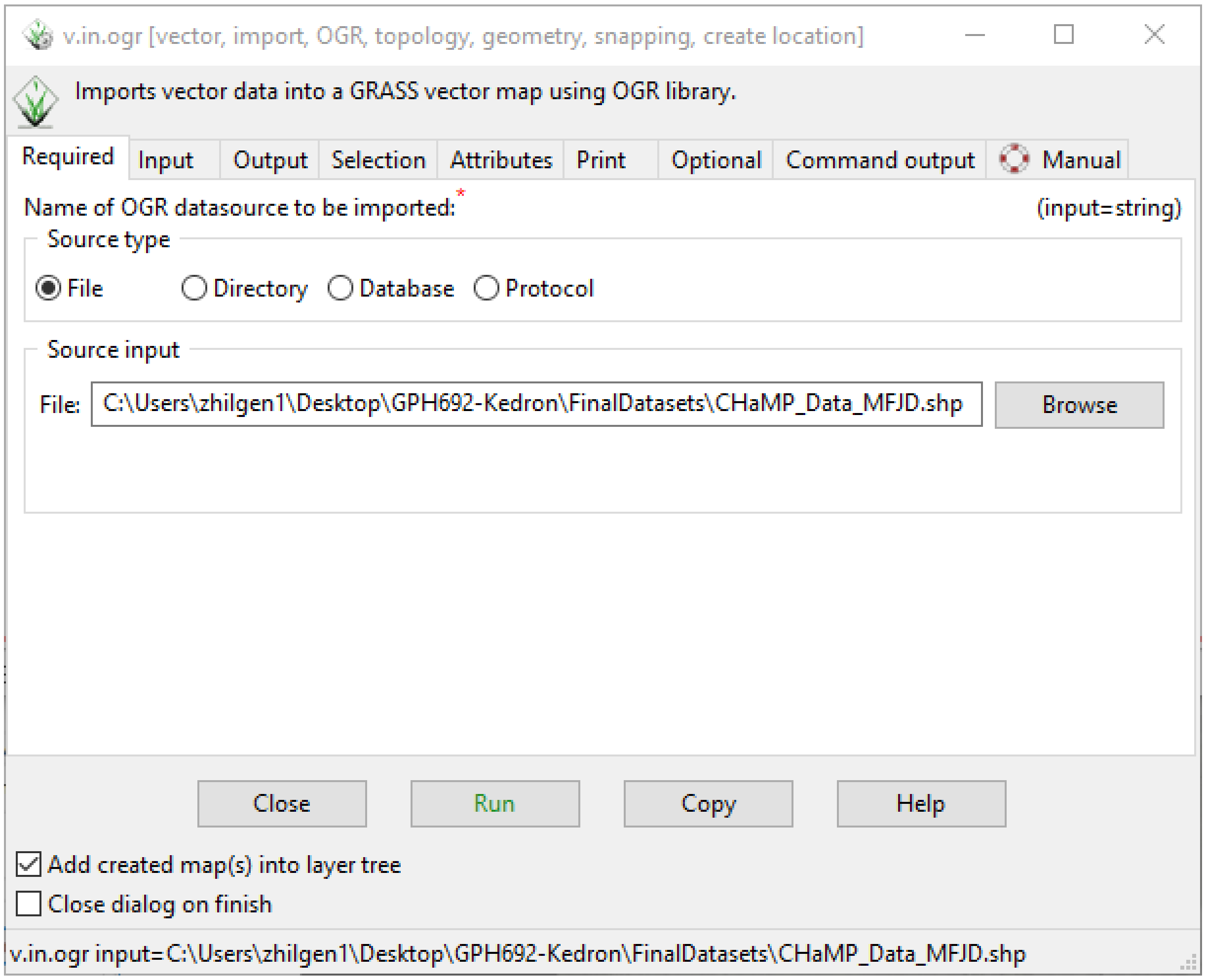
☐ Override projection check (use current location's projection)
☐ Allow output files to overwrite existing files
☑ Add imported layers into layer tree
☐ Close dialog on finish

Close Import

Source settings | Import settings

Figure 6. The raster import window.

If you chose not to import the CHaMPS_Data_MFJD shapefile when you created the location, the following steps will show you how to do so. In the File menu or the Modules tab, select the "**Import of common vector formats** [**v.in.ogr**]" tool. This will open a window that lets you browse for the shapefile you want to import (Figure 7). Run the tool and your vector will import! Make sure to have "File" selected in the "Name of OGR datasource to be imported" box.



Imports vector data into a GRASS vector map using OGR library.

Name of OGR datasource to be imported:* (input=string)

Source type: [✓ File] Directory Database Protocol

Source input

File: C:\Users\zhilgen1\Desktop\GPH692-Kedron\FinalDatasets\CHaMP_Data_MFJD.shp

[✓] Add created map(s) into layer tree

[ ] Close dialog on finish

```
v.in.ogr input=C:\Users\zhilgen1\Desktop\GPH692-Kedron\FinalDatasets\CHaMP_Data_MFJD.shp
```

Figure 7. The vector import window.

Summary

This short handout demonstrated how to create a new Location for use in GRASS GIS. It also taught how to import new raster and vector data. Future handouts will explain how to view that data and process it further.

2 Digitizing in GRASS

Purpose: One of the many great things we can use GRASS GIS for is to digitize, collapse, and extract longitudinal profiles from a stream. We can use these same tools to create a valley centerline or extract valley cross sectional profiles. This handout will walk through the tools and methods to extract longitudinal profiles, calculate stream length and valley centerline length, and create cross-sectional profiles of the stream valley. These outputs will be used, directly, as inputs into the Rosgen Classification table for slope and sinuosity. To learn more about these concepts, please refer to the recorded YouTube videos

Sinuosity: https://youtu.be/rVs4MSw1e5s
Slope/Longitudinal Profile: https://youtu.be/SK2BFg2PmlQ

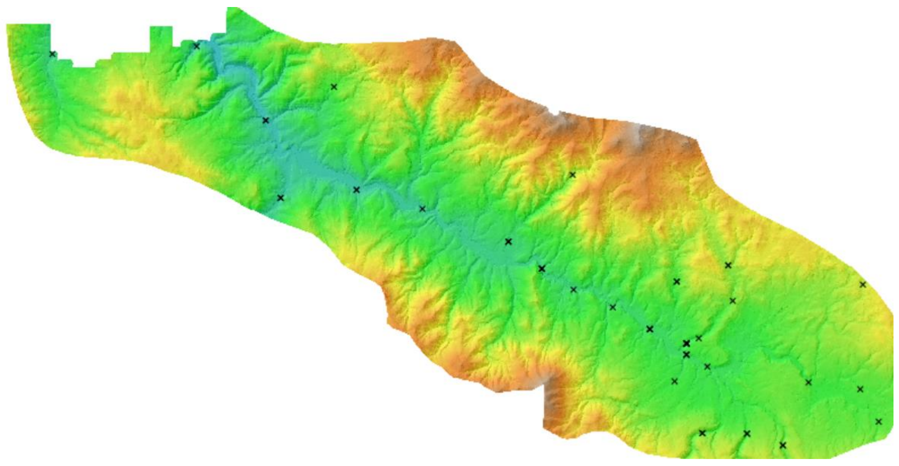
Create a map for stream analysis

In previous walkthroughs, we have gone over how to build locations and mapsets in GRASS GIS and how to load those into a map viewer. Please refer to those if you need a refresher. We will start with the **John Day digital elevation model (DEM)** and **hillshade** loaded into the viewer. Set the John Day DEM **opacity** to **65%,** displayed on top of the hillshade. Load the **CHaMP_Data** to the top of the map. (See map style below)

Hints for visualizing data in GRASS:

- Add data to the map: double-click the dataset in the Data tab
- Reorder layers: click and drag layers on the layers tab
- Change raster colors: right click the layer and **set color table interactively** (the example below uses the **elevation** color table)
- Change layer transparency: right click the layer and **change opacity level**
- Open a vector layer attribute table: right click the layer and **show attribute table**
- Highlight a selected feature: double-click the feature's row in the attribute table

To save your map visualization work, go to **File → Workspace → Save**
When you reopen GRASS, you can similarly load previous workspaces at **File → Workspace → Open**



Map style for stream digitization

Digitize Stream Banks and Valley Sides

Our first goal will be to digitize the **river banks** and the **edges of the valley**.
Let’s navigate to the a specific CHaMP location in order to digitize channel morphology. We will be using a site on the Middle Fork of the John Day River (Site: CBW05583-275954) for our example.

We’ll use the following workflow for digitizing both the stream banks and the valley outline. Digitizing practices typically suggest digitizing the same features multiple times, to ensure consistency (Micheli et al., 2004). Sometimes, when constrained by resolution or with features that are tough to define, this practice can help delineate a range of possible extents and distinguish areas with greater uncertainty. We will observe the following standards:

1) All digitized features will be digitized three times.
2) All features will be digitized at a scale of 1:2000. (set this with the status bar at the bottom of the map: change Coordinates to Map Scale and type the scale in at left)
3) Name layers clearly and iteratively, for clarity (Banks1/2/3, Valley1/2/3).
   a) Each layer will have two line features: the two sides of the river bank or the two sides of the valley.
4) Alternate between creating features (Banks1, Valley1, Banks2, Valley2, Banks3, Valley3) to reduce likelihood that “muscle memory” will skew outputs.

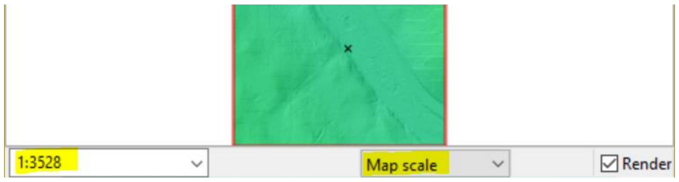


Figure 1b: Use the status bar to change the map scale.

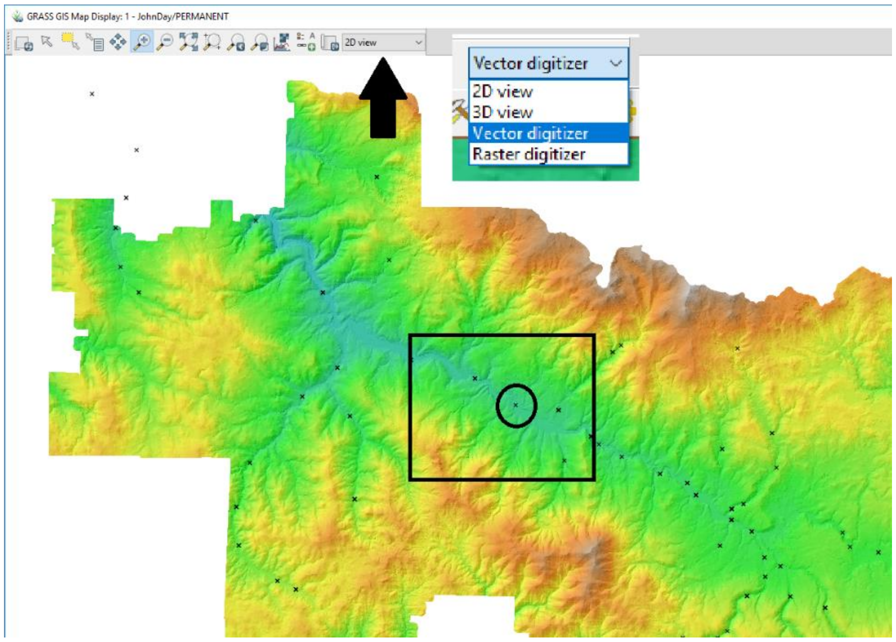


Figure 1c. Extent of the John Day raster dataset with CHaMP data points. The black rectangle is outlining the computational extent, while the circle shows the workflow example data point. The black arrow is indicating the location of the digitizer option (shown in the callout box).

We first need to set up the **vector digitization** window (Figure 1b).
A new toolbar will appear in the Map Display window with options for vector creation and editing. Select "**New vector map**" (Figure 2) at the left end of the digitizing toolbar and a new window will open (Figure 3a). Once you have created the vector a second window will open (Figure 3b), where you can manage the attribute table for the new vector layer. The first time through, create a **Banks1** layer with an additional field, **Length,** of type **Double**.

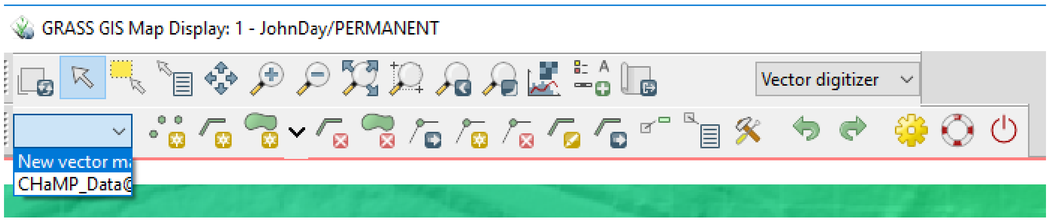
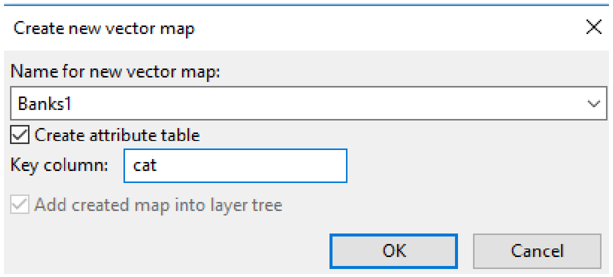


Figure 2. Vector digitizer toolbar and the dropdown menu for selecting which vector to edit/create a new vector.



Create new vector map

Name for new vector map:

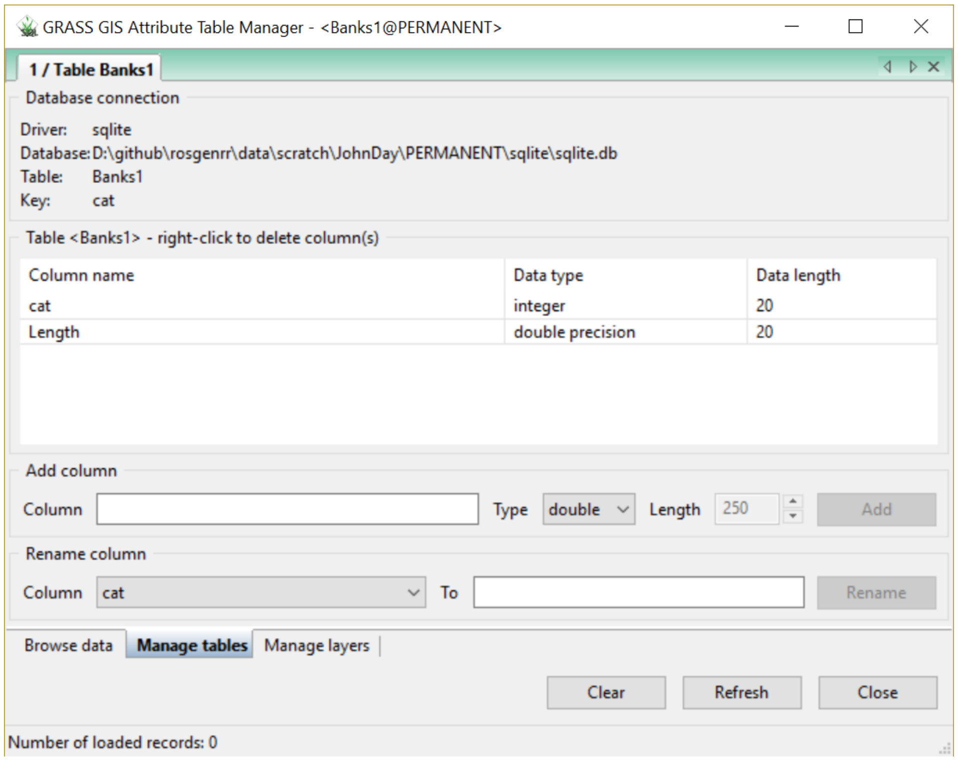
Banks1

[✓] Create attribute table

Key column: cat

[✓] Add created map into layer tree

Figure 3a. New vector map windows. Window that opens when the New Vector Map option is selected. Here, enter a name for your new vector map, starting with "Banks1".



GRASS GIS Attribute Table Manager - <Banks1@PERMANENT>

1 / Table Banks1

Database connection

Driver: sqlite
Database: D:\github\rosgenrr\data\scratch\JohnDay\PERMANENT\sqlite\sqlite.db
Table: Banks1
Key: cat

Table <Banks1> - right-click to delete column(s)

Column name	Data type	Data length
cat	integer	20
Length	double precision	20

Add column

Column: Type: double Length: 250

Rename column

Column: cat To:

Number of loaded records: 0

Figure 3b. Window that opens after the vector has been created in A.

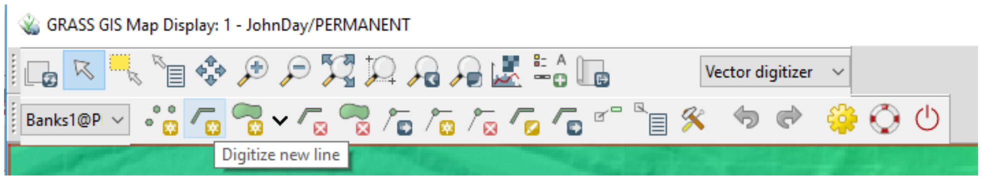


Figure 4. Location of the “Digitize new line” tool. Make sure that the layer you want to be adding a line to is selected in the left dropdown menu.

After creating the vector, select the “Digitize new line” tool (Figure 4). There are many different methods for digitizing streambanks, that depend on the available dataset. To maintain consistency, we will utilize the John Day DEM rather than a satellite image. The cells of this raster dataset have a spatial resolution of 1 meter. This is an elevation-based dataset, so we do not have to worry about vegetation obstructing our vision. However, resolution and color ramps can impact how we interpret the elevation models. If we were digitizing a satellite image or an orthomosaic, we may be able to see the streambanks, but they could be covered by vegetation or flooded water. We will not have to deal with it in this dataset, but there are some examples of the different ways to digitize orthomosaic imagery in the Appendix.

For the channel banks, we will want look for distinct breaks in slope around the channel or obvious slopes in the hillshade. We’ll want to be careful! The road on the north side of the valley looks really similar to a river, but we can see that it is raised above the valley floor. It can be a bit tough to make out the edges of the river, but with careful digitizing at a consistent scale, we can get a good idea of the outline of the channel (Figure 5B). Rosgen (1994) suggests encompassing a distance of 20 channel widths (10 upstream, 10 downstream). In the example, the stream width was about 15 m wide, near the collection point. I digitized about 250 m upstream and downstream, with the sampling location datapoint in the center (Figure 5B), so I extended it a bit more than necessary. There is a measure tool in the **Analyze map** menu:

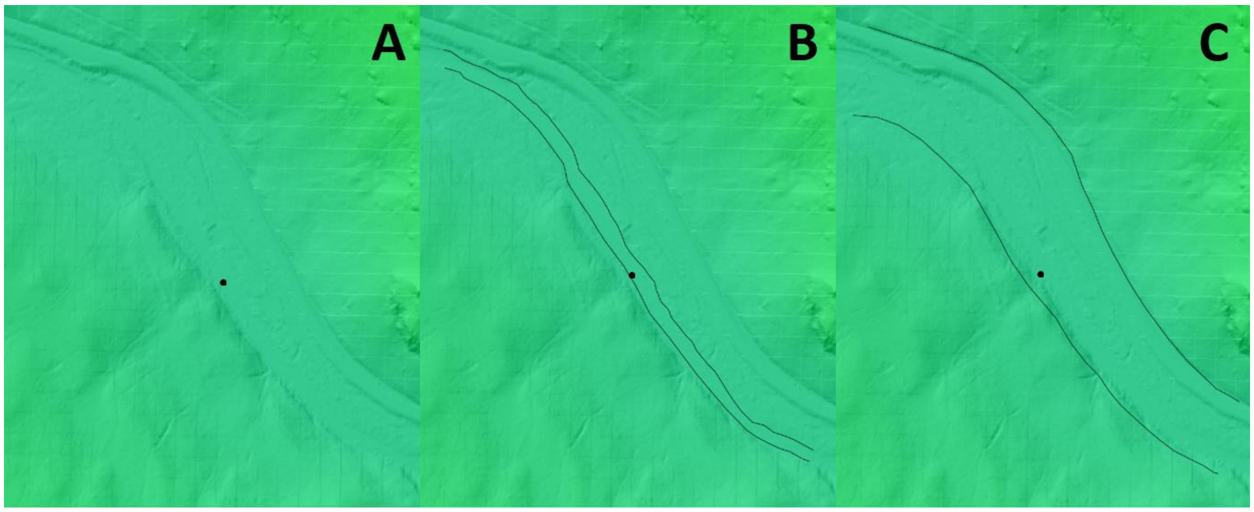
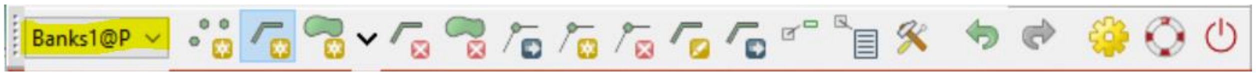


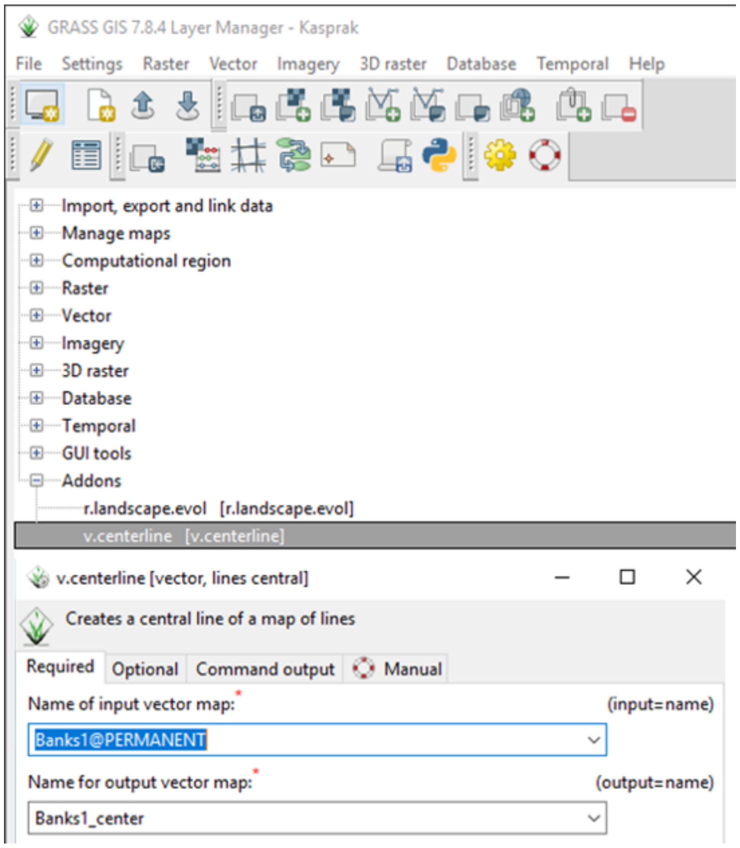
Figure 5. A) DEM of the Middle Fork of the John Day River with the example datapoint in the center of the map. B) An example of the digitized streambank lines from our first digitized banks. C) An example of the digitized valley extent from our first dataset.

The methods for digitizing the valley will be similar. We want to capture the extent of the stream valley and cover the same distance as we did with the banklines (Figure 5C). We can see some obvious breaks in slope, evident by the underlying hillshade. Sometimes, it is a bit tougher to tell, but we mostly want to get the general shape of the valley floor so that our valley centerline is accurately represented. We'll repeat this process three times, for both the streambanks and the valley boundaries. Once we've finished that, we can move on to creating our centerlines. Hint: create additional new layer by using the drop-down menu at the left side of the digitizing toolbar:



Create centerlines

Now that we have our digitized banklines and valley boundaries, the next task is to generate a centerline. To do this, we'll first need to go to the ***Settings* → *Addons extensions* → *Install extensions from addons* [g.extension]** in the Layer Manager window. That will open a menu in which you can install and manage extensions within GRASS. Enter "**v.centerline**" in the search bar and double-click on it to install the extension. You can find the extension in the **Modules** tab in the Layer Manager window, under the Addons (expand the plus sign) (Figure 6). You may need to close and restart GRASS in order to see the Addon.


GRASS GIS 7.8.4 Layer Manager - Kasprak
File
Settings
Raster
Vector
Imagery
3D raster
Database
Temporal
Help
Import, export and link data
Manage maps
Computational region
Raster
Vector
Imagery
3D raster
Database
Temporal
GUI tools
Addons
r.landscape.evol [r.landscape.evol]
v.centerline [v.centerline]
v.centerline [vector, lines central]
Creates a central line of a map of lines
Required
Optional
Command output
Manual

Name of input vector map:* (input=name)
Banks1@PERMANENT
Name for output vector map:* (output=name)
Banks1_center

Figure 6. The Modules tab of the Layer Manager window with the "v.centerline" tool highlighted and the tool menu opened. The menu is where the input vector map (either the Banks1/2/3 or Valley1/2/3) will be selected and the output name will be input.

Once you open the **v.centerline** module, all that needs to be input is the chosen vector map (Banks1/2/3 or Valley1/2/3) and the desired name for the output vector map. Maintaining a consistent naming convention is a good way to keep track of progress. For example, adding "_centerline" to the end of your original names (i.e. **Banks1_centerline**), ensures that you know what step in the process you are at. Running this tool will give you a single line vector output that represents the calculated midpoint line between the two digitized lines you created to represent the banks or the valley (Figure 7.)

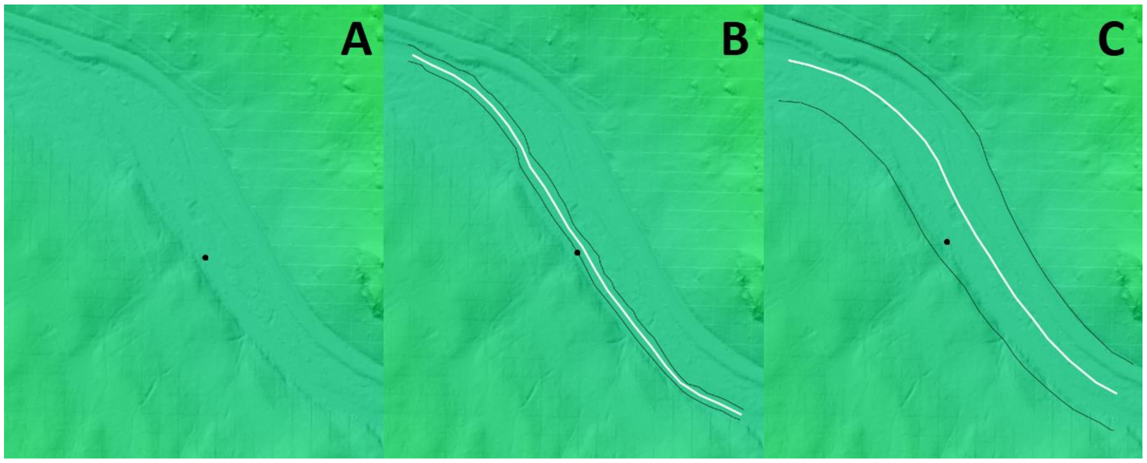


Figure 7. A) DEM of the Middle Fork of the John Day River with the example datapoint in the center of the map. B) An example of the digitized streambank lines (in black) and the related centerline (in white). C) An example of the digitized valley extent (in black) and the related centerline (in white).

Calculate Line Lengths

Once you have six centerlines: three versions of a river bank centerline and three versions of the valley centerline, then it's time to calculate their **lengths**, and finally sinuosity.

In GRASS, calculating a feature's length will be a whopping three steps:

1) Assign **cat** (category) numbers to each of the center lines with the **v.centerline** tool.
   a) The **name of the input vector map** is your center line vector.
   b) The action to be done is **add**
   c) In the **Optional tab**, enter the **name for output vector map.** This could be your centr line layer name plus a "_cat" suffix.
   d) The **category value** is 1, with an **increment** of 1
   e) Note: this step is essentially assigning ID's to the feature(s) so that they can be attached to an attribute table.

2) Create an attribute table with the "**v.db.addtable**" tool. Choose the vector map that you want to add a table to (either the Banks or Valley vectors) and simply run the tool.
   a) In the **Definition** tab, fill in the **name and type of the new column(s)**, option with "***length double***" to name the column "Length" and assign it a double precision decimal data type.

3) Calculate length with the "**v.to.db**" tool, which we can find by searching, or in the Modules tab → Vector → Update database values from vector [v.to.db] (Figure 9). Follow the set up shown in Figure 9, where:
   a) the **name of the vector map** is the vector you are currently editing (start with Banks1_centerline)
   b) the **value to upload** is **length** (as in the function)
   c) the **name of attribute column(s)** is also **length** (as in the attribute created above)
   d) in the Optional tab:
      i) allow output files to overwrite existing files and
      ii) set the Units to meters

Summary

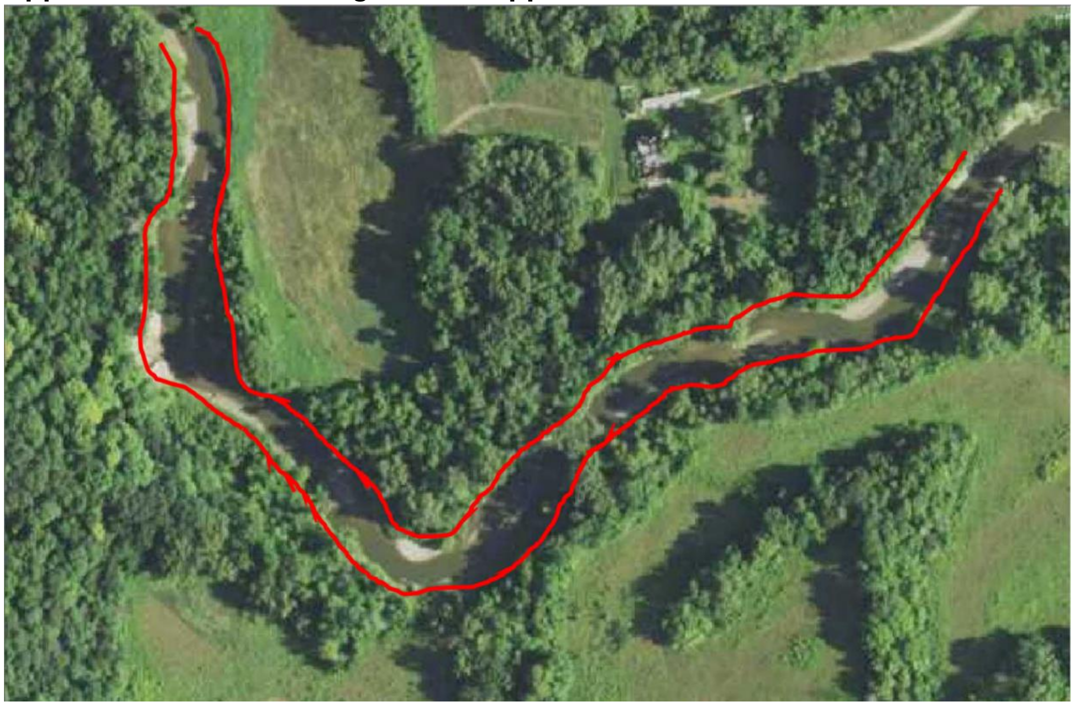
You’ve finished with this section when you have calculated six lengths: stream bank centerline length for three bank digitization attempts and valley centerline lengths for three valley edge digitization attempts.
In this handout, we talked about how to digitize lines and the methods for digitizing them in GRASS GIS. We also learned how to install add-on packages and use the v.centerline tool to collapse lines to a single average line. These skills will proof useful on the next handout, where we learn how to use those centerlines to extract and graph raster cells that intersect our lines, to calculate slope values to input into our classification scheme!

References

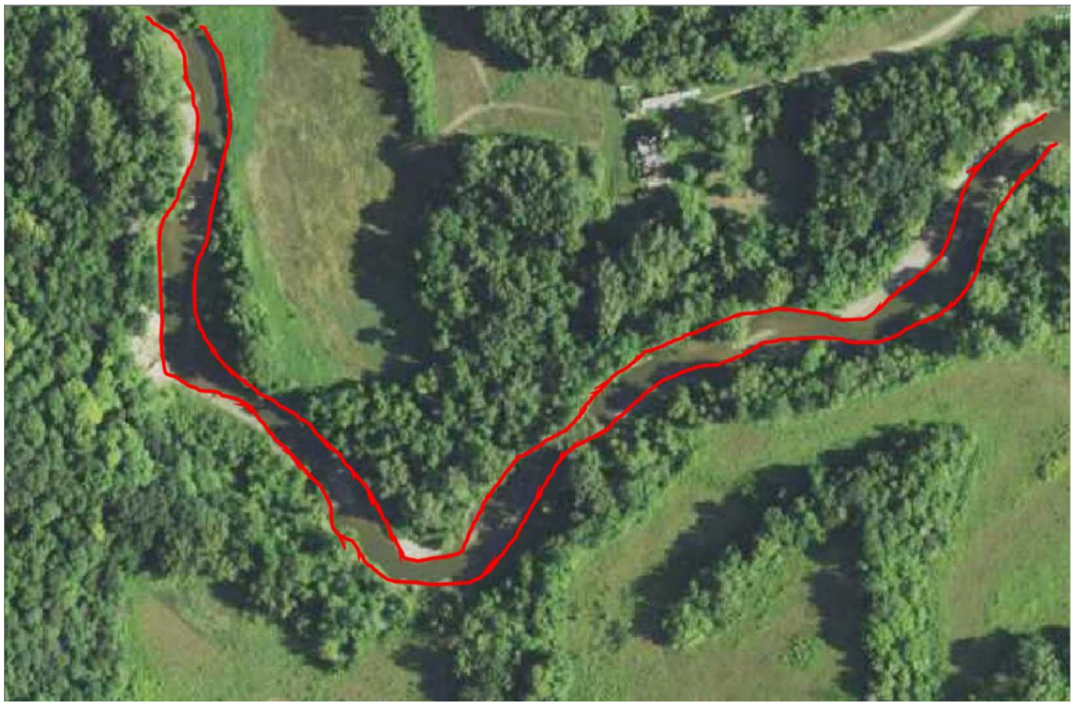
Micheli, E. R., J. W. Kirchner, and E. W. Larsen. "Quantifying the effect of riparian forest versus agricultural vegetation on river meander migration rates, Central Sacramento River, California, USA." *River research and applications* 20, no. 5 (2004): 537-548.

Rosgen, D.L., 1994, A classification of natural rivers: CATENA, v. 22, p. 169–199, doi:10.1016/0341-8162(94)90001-9.

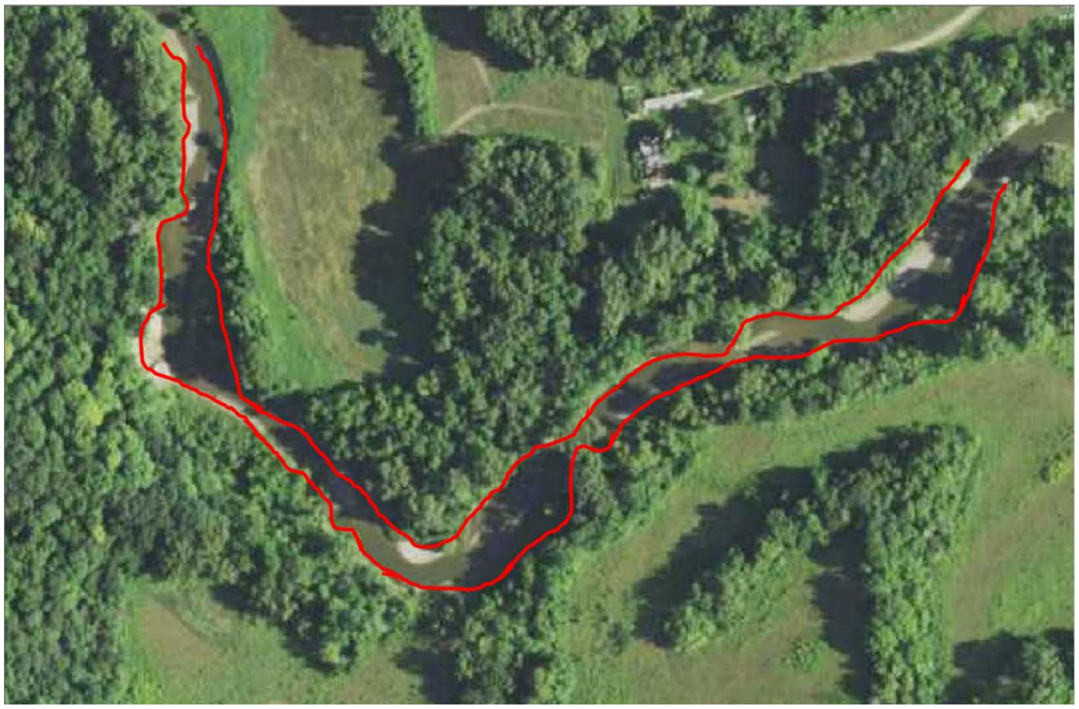
Appendix: Other Bank Digitization Approaches



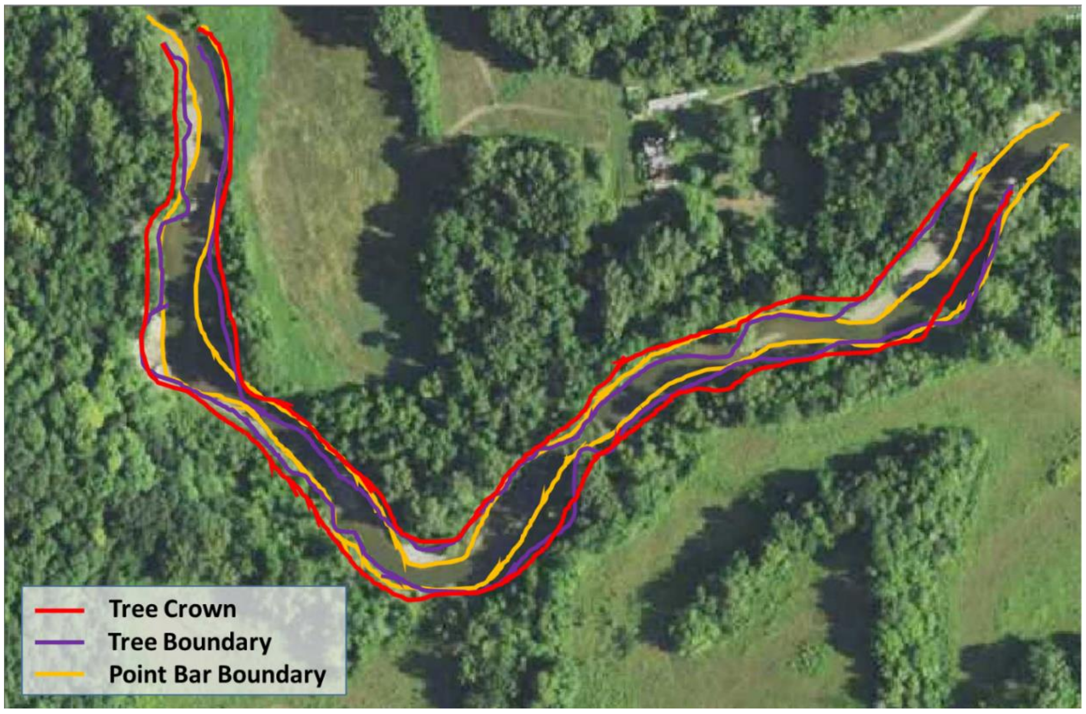
This method of bankline digitization utilizes the crown (middle) of the trees that surround the river to identify a reasonable bankline. While it is possible that this method overestimates the width, it is likely that the surrounding trees are overhanging the river, so the crown is likely near the bankline.



This method digitizes utilizes point bars, depositional features on the inside of a river meander bend, to approximate the banklines. In low flow situations, this is reasonable, but may increase sinuosity by adding curvature to the river.



This method digitizes the boundary of the treeline, as we do not technically know where the bank exists underneath the tree and this focuses more on the water we can see. Notice how "lumpy" the boundary lines look here. You can imagine this may add unnecessary distance to the end product.



Here are all three methods are shown on top of one another. The preferred and suggested method is to use the crown of the tree when the bank is not obviously visible. This provides the most reasonable approximation of the bankline, when not visible.

3 Visualize River Profiles

Purpose**:** The purpose of this handout is to take the methods from the GRASS GIS digitization handout and employ them to extract data from raster cells and put them into a two-dimensional view to calculate slope or to better understand the shape of the valley. If you are in need of a refresher, refer back to the digitization workflow, as this picks up where that one left off. The concepts to consider in this lab are best portrayed by the following YouTube videos:

Sinuosity: https://youtu.be/rVs4MSw1e5s
Slope/Longitudinal Profile: https://youtu.be/SK2BFg2PmlQ

In the previous walkthroughs, we went over how to build locations and mapsets, and then how to digitize vector maps, add packages, and use new tools, such as the *v.centerline* tool, using GRASS GIS. The primary purpose of this walkthrough is to teach you how to extract raster cells, along a line, to plot, view, and analyze the data in your preferred plotting software (i.e. Microsoft Excel, Google Sheets, R, Matlab, etc.).

We will start off where the digitization handout concluded. Make sure you have the John Day DEM, Banks_Center vectors, Valley_Center vectors, and CHaMP data vector loaded into the Map Display.

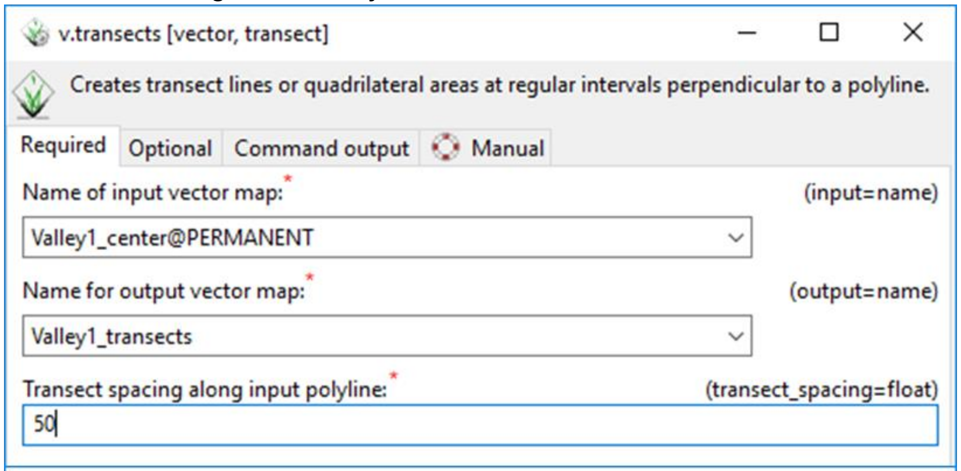
Build Transects

Cross sectional transects can be a really useful tool to examine the shape of a river valley as the river advances downstream. The shape of the valley can possibly tell us about the evolutionary history of the river, its potential for flooding, and even the potential threat of natural hazards, such as mass wasting events.

Add the transects tool to GRASS by going to **Settings → Addons extensions → install extension from addons**. Search for the **v.transects** tool and **install** it.
In the Modules tab, found in the Layers Manager window, search for the "**v.transects"** tool (Figure 1). For this tool, we will select the valley centerline vector we created.

> Recall that we created three different valley boundary datasets, so we have three separate valley centerlines. You could combine these, if you would like, and run the **v.centerline** tool again to get an average valley centerline. To do so, you would start a vector editing session, create a new vector, and copy→paste the centerlines from your three valley centerline vectors. Then you would run the v.centerline tool again. While this is not necessary, it may be useful if you have high variability in your centerline vectors.

Back in the v.transects tool, we will input our desired vector map, as well as a name for the output map. Then, we will choose a desired spacing between transects. This tool will generate transects along the line at whatever spacing interval we define, and will depend on the desired output. In this example, I chose a spacing of 50 m. Note that you can enter a unitless number here, as most tools that require distance inputs reference the default map unit if none are provided. Now switch to the optional tab. We need to specify the distance we want the transect lines to extend to either side of the line. In the example, I chose a distance of 250 m to either side. Make sure to also have the "Allow output files to overwrite existing files," as you may want to try different distances to get your preferred coverage. You will likely enter the same value for both sides, but, should you want to finetune these numbers, you should be able to click on your vector in the Map Display window and see arrows pointing in the direction the vector is oriented. Finally, make sure the input feature type, how your transect spacing is measured, and which line is the transect perpendicular to (the dropdown menus at the bottom of the Optional tab) are all set up how I have it in Figure 1. The you can **Run** the tool.

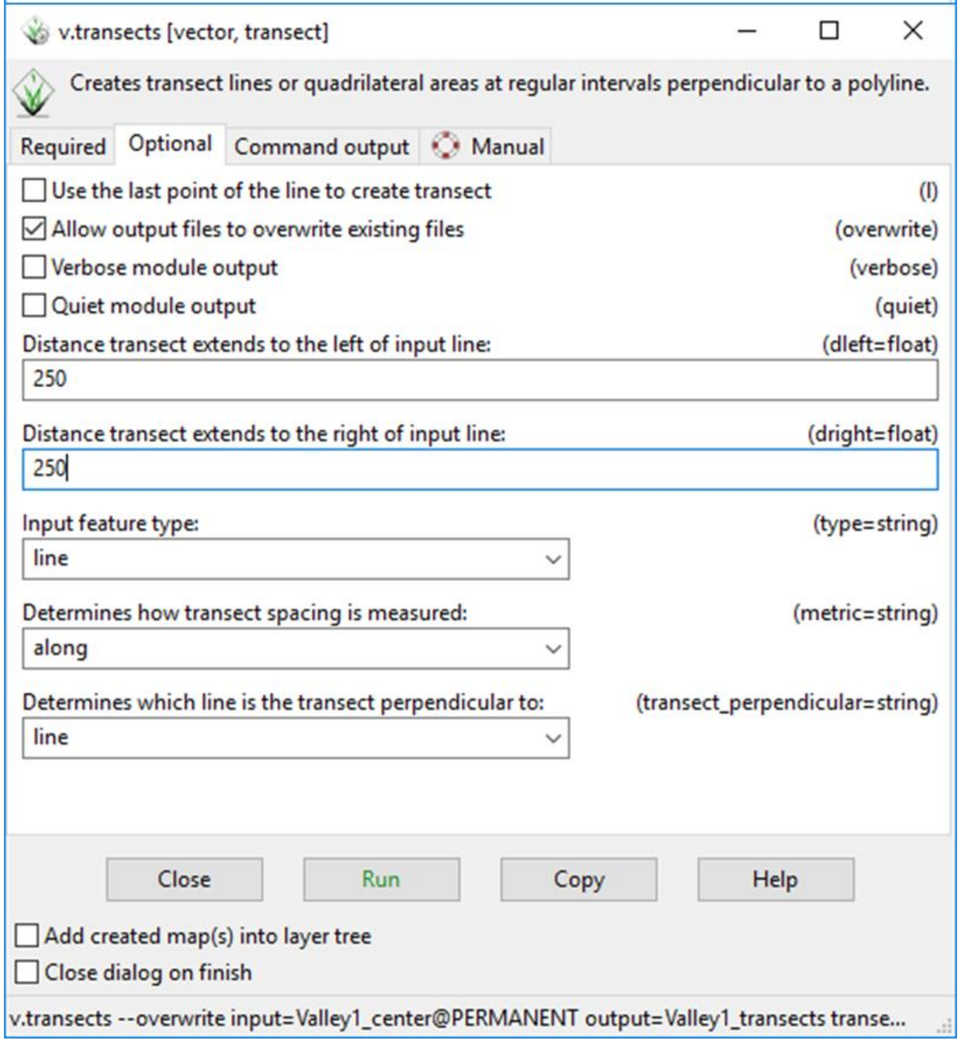

v.transects [vector, transect]

Creates transect lines or quadrilateral areas at regular intervals perpendicular to a polyline.

Name of input vector map:* (input=name)
Valley1_center@PERMANENT

Name for output vector map:* (output=name)
Valley1_transects

Transect spacing along input polyline:* (transect_spacing=float)
50


v.transects [vector, transect]

Creates transect lines or quadrilateral areas at regular intervals perpendicular to a polyline.

☐ Use the last point of the line to create transect (l)
☑ Allow output files to overwrite existing files (overwrite)
☐ Verbose module output (verbose)
☐ Quiet module output (quiet)

Distance transect extends to the left of input line: (dleft=float)
250

Distance transect extends to the right of input line: (dright=float)
250

Input feature type: (type=string)
line

Determines how transect spacing is measured: (metric=string)
along

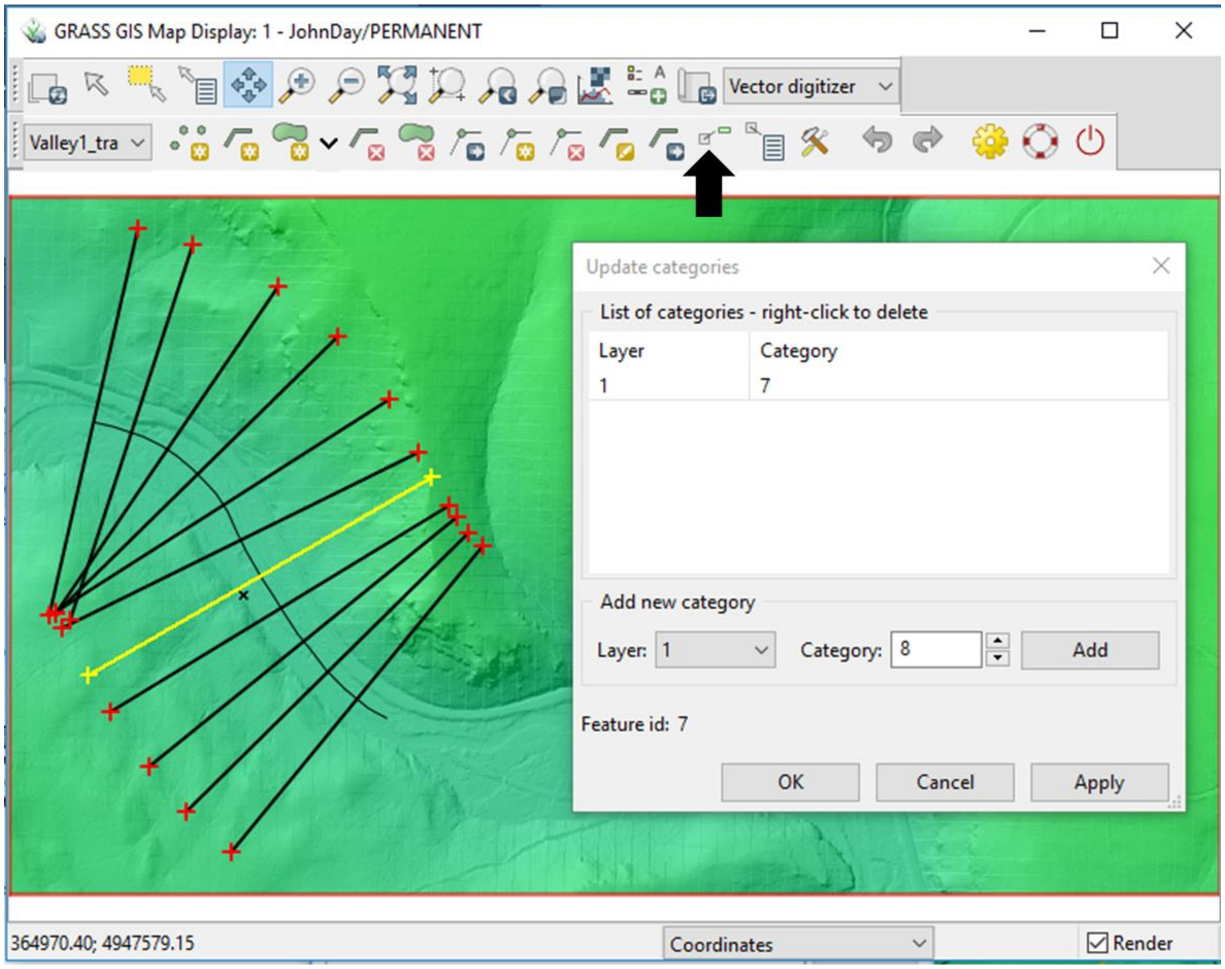
Determines which line is the transect perpendicular to: (transect_perpendicular=string)
line

☐ Add created map(s) into layer tree
☐ Close dialog on finish

```
v.transects --overwrite input=Valley1_center@PERMANENT output=Valley1_transects transe...
```

Figure 1. Transect generation tool window. The top window shows the Required tab, while the bottom window shows the Optional tab. Make sure to allow overwrite, so that you can quickly rerun the tool if you do not get the desired output.

The output from the v.transects tool will draw as many lines as will fit along the input line, starting with a transect at the initial vertex of the vector. Fortunately, we have a CHaMP data point close to one of the new transects! We will choose that one for further analyses. In the Map Display window, change to the vector digitizer, make sure the newly created transect feature is selected in the vector menu, and click on the Display/update categories tool, shown by the black arrow in Figure 2. Click on the transect closest to your CHaMP data point to see what the ID of the line is. You can see that the selected line is listed as Category = 7. This will be our reference when we extract that line from the dataset.



Layer	Category
1	7

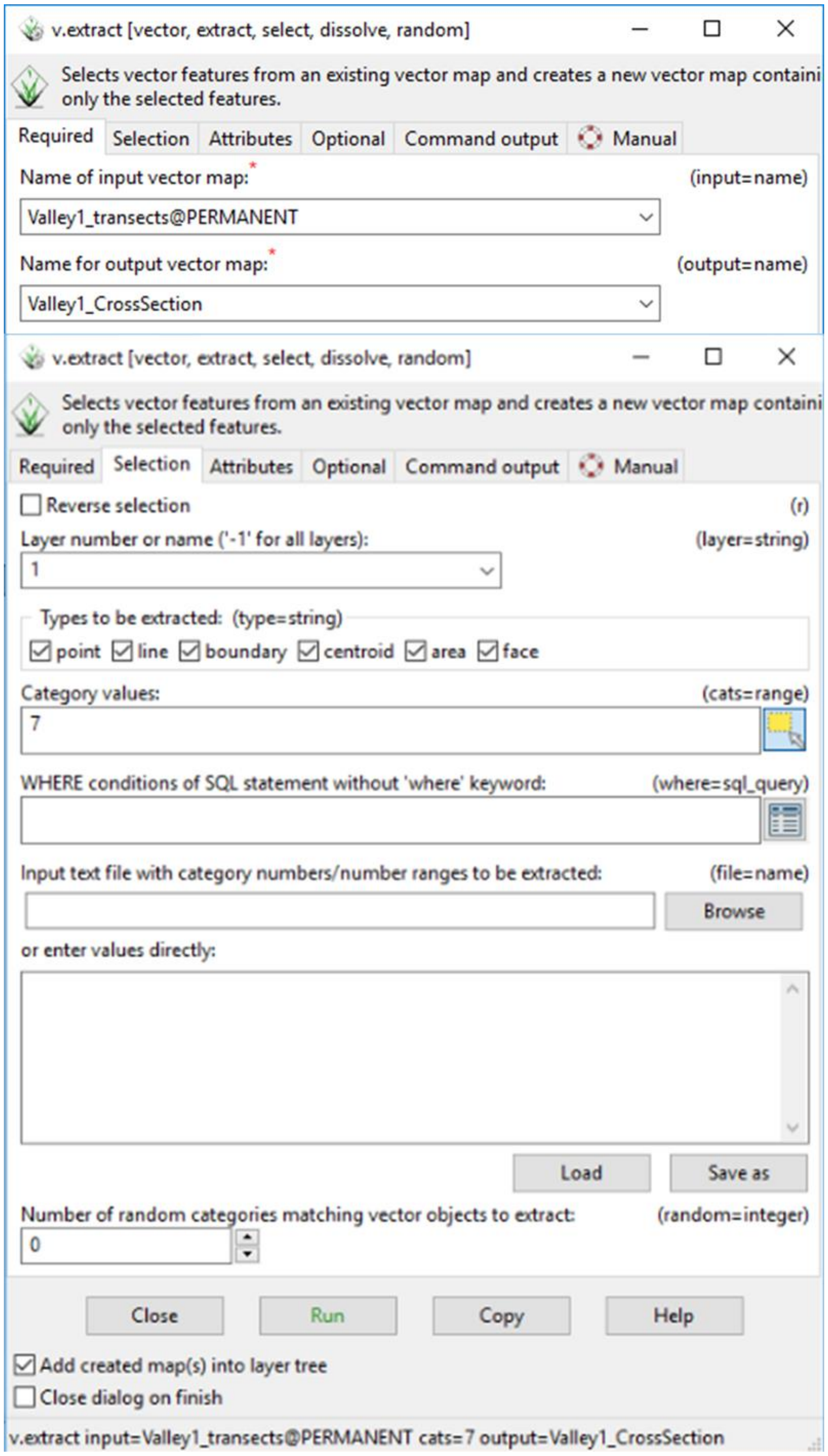
Layer: 1 Category: 8

Feature id: 7

364970.40; 4947579.15

Figure 2. Output from the v.transect tool. Lines are drawn perpendicular to the input line, which is the valley centerline in this example. The black arrow shows the location of the Display/update categories tool. The Update categories window shows the vector feature category as **7**, which is unique to this line.

Next, flip back to the Modules tab and search for the **"v.extract"** tool. This tool lets you extract features from a vector dataset that meet a listed criteria. Follow the specifications in Figure 3 to set up your extraction, personalizing it for your dataset. The Selection tab allows you to list Category values to extract. In our example case, we will input the value 7, which we found with the Update/display categories tool. Run the tool.



v.extract [vector, extract, select, dissolve, random]

Selects vector features from an existing vector map and creates a new vector map containi only the selected features.

Name of input vector map:* (input=name)
Valley1_transects@PERMANENT

Name for output vector map:* (output=name)
Valley1_CrossSection

v.extract [vector, extract, select, dissolve, random]

Selects vector features from an existing vector map and creates a new vector map containi only the selected features.

☐ Reverse selection (r)

Layer number or name ('-1' for all layers): (layer=string)
1

Types to be extracted: (type=string)
☑ point ☑ line ☑ boundary ☑ centroid ☑ area ☑ face

Category values: (cats=range)
7

WHERE conditions of SQL statement without 'where' keyword: (where=sql_query)

Input text file with category numbers/number ranges to be extracted: (file=name)

or enter values directly:

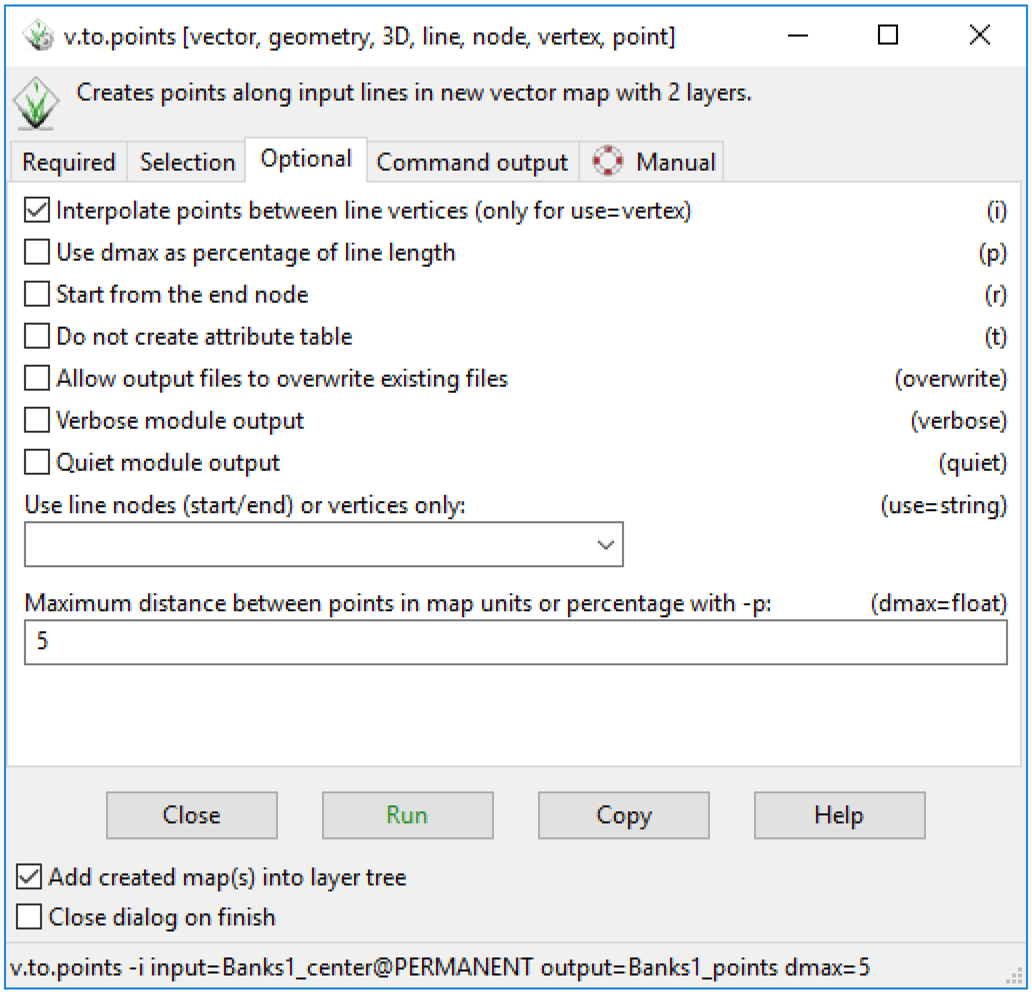
Number of random categories matching vector objects to extract: (random=integer)
0

☑ Add created map(s) into layer tree
☐ Close dialog on finish

```
v.extract input=Valley1_transects@PERMANENT cats=7 output=Valley1_CrossSection
```

Figure 3. The vector extraction tool window. Like most of the other tools we have discussed, select the input variable and output name in the Required tab. Switching to the Selection tab, we can see just how extensive this tool is and just how much we can do to specify our selections.

Now that we have a single transect line to extract, we need to prepare that line for raster cell extract. This is where the goals of the two desired outputs in the handout converge. Recall that we want to look at the slope of the reach of river we are examining for the classification. We already have the bank centerline(s). With the valley cross sectional transect and the bank centerlines, we want to extract the raster cells that overlay, to plot and analyze in other software. The easiest way to do this is to use the lines to generate points at a specific density. Most GIS software has a way to do this. In GRASS, the **"v.to.points"** is the tool to use for this method (Figure 4). Search that tool in the Modules tab. In the Required tab, input the extracted variable and the desired output name. In the Optional tab, select the Interpolate points between line vertices option and input the maximum distance between points. The value you input here is up to you, but consider what the purpose is going to be. The resolution of our John Day LiDAR DEM is 1 m. You could input 1 as your maximum distance, to try and capture every cell that the lines intersect. This will result in a bulkier, but continuous dataset for future steps.


v.to.points [vector, geometry, 3D, line, node, vertex, point]

Creates points along input lines in new vector map with 2 layers.

[✓] Interpolate points between line vertices (only for use=vertex) (i)
[ ] Use dmax as percentage of line length (p)
[ ] Start from the end node (r)
[ ] Do not create attribute table (t)
[ ] Allow output files to overwrite existing files (overwrite)
[ ] Verbose module output (verbose)
[ ] Quiet module output (quiet)

Use line nodes (start/end) or vertices only: (use=string)

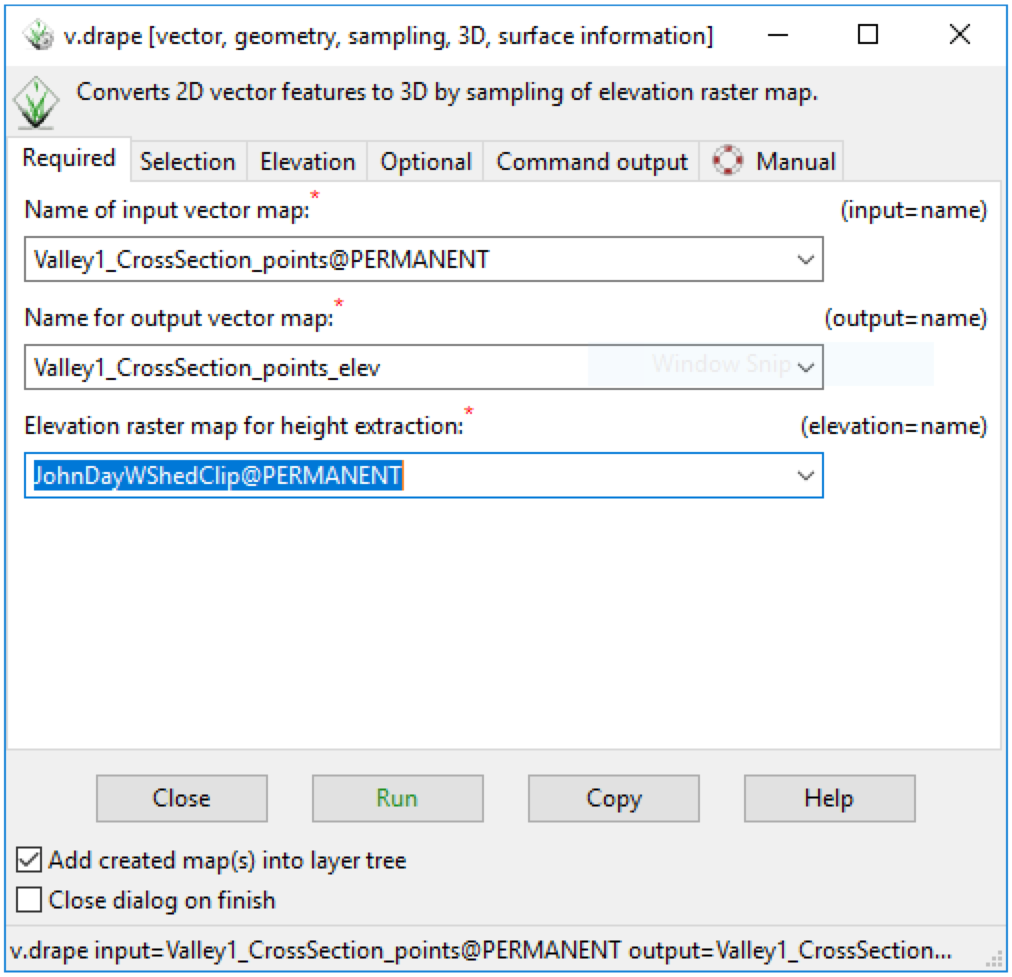
Maximum distance between points in map units or percentage with -p: (dmax=float)
5

[✓] Add created map(s) into layer tree
[ ] Close dialog on finish

```
v.to.points -i input=Banks1_center@PERMANENT output=Banks1_points dmax=5
```

Figure 4. The specification window for the tool that generates points along a vector map. Make sure to create these points for your bank (river) centerlines and valley transect.

We are almost there! Using the **“v.drape”** tool, we can choose an input vector map to extract points from an elevation dataset (Figure 5). This is a pretty straightforward tool, for our purposes. Just input the input variable, output name, and elevation map and run the tool. The output will duplicate the input points, but append the attribute table with the height values for the cell that each point falls on.


v.drape [vector, geometry, sampling, 3D, surface information]

Converts 2D vector features to 3D by sampling of elevation raster map.

Name of input vector map:* (input=name)
Valley1_CrossSection_points@PERMANENT

Name for output vector map:* (output=name)
Valley1_CrossSection_points_elev

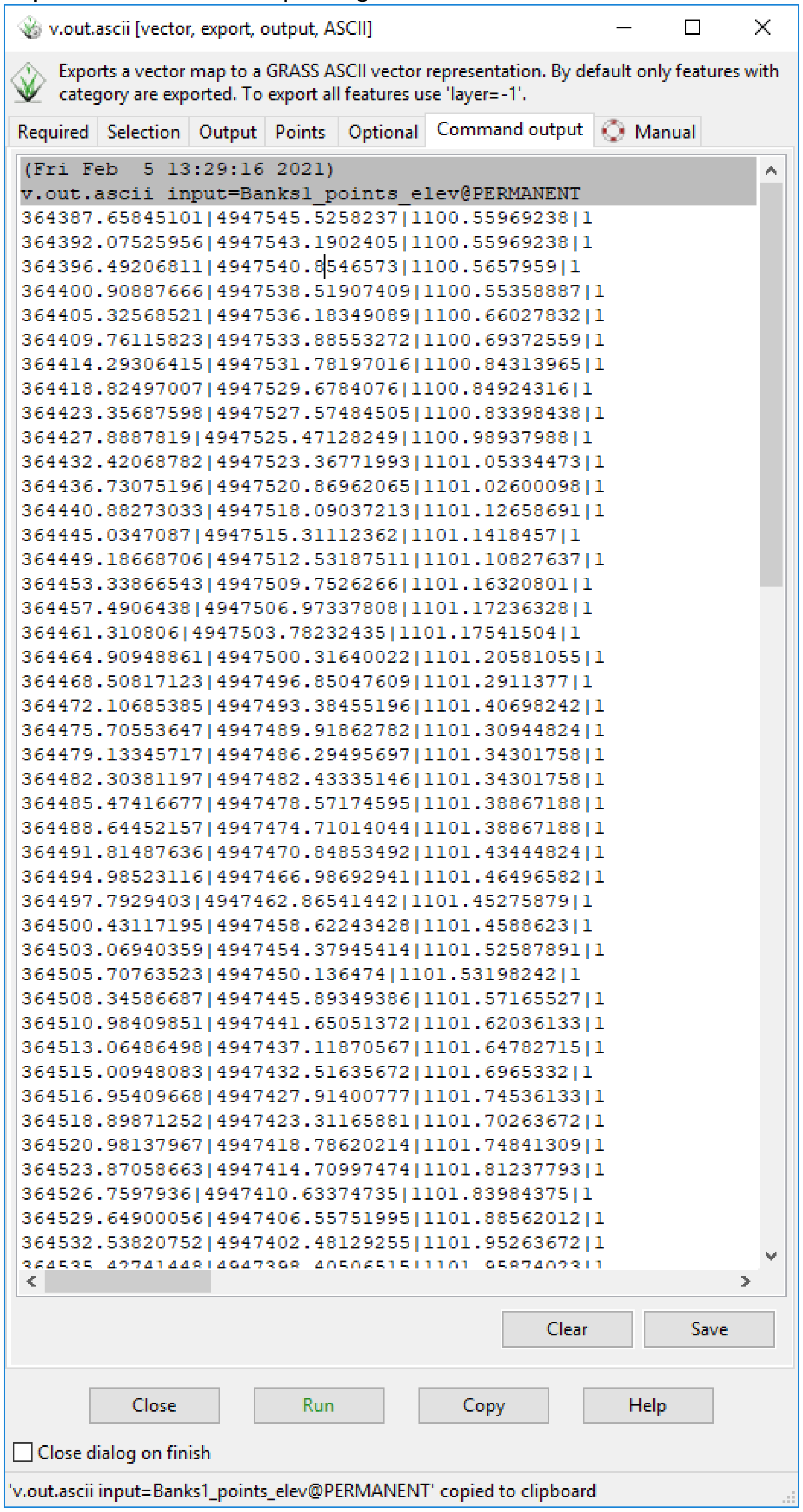
Elevation raster map for height extraction:* (elevation=name)
JohnDayWShedClip@PERMANENT

☑ Add created map(s) into layer tree
☐ Close dialog on finish

```
v.drape input=Valley1_CrossSection_points@PERMANENT output=Valley1_CrossSection...
```

Figure 5. The input window for the v.drape tool. This is an easy tool to run, for our purposes, only requiring the point vector map we previously created and the elevation raster from which to extract the values from.

The final tool we need to use in GRASS is the **“v.out.ascii:** tool, which creates an output list of the attribute table for an input vector map (Figure 6). For this tool, select the points that were created with the v.drape tool (which have the elevation values we want). Run the tool and change to the Command Output tab. Copy the list that was generated. Paste it into a TEXT (.txt) file and save the file. Alternatively, you can go to the output tab and choose an output location and include a “.txt” at the end of your file name to save it as a text file. Now we are ready to import into our favorite plotting software and take a look at the data in a two-dimensional format.



Exports a vector map to a GRASS ASCII vector representation. By default only features with category are exported. To export all features use 'layer=-1'.

```
(Fri Feb  5 13:29:16 2021)
v.out.ascii input=Banks1_points_elev@PERMANENT
364387.65845101|4947545.5258237|1100.55969238|1
364392.07525956|4947543.1902405|1100.55969238|1
364396.49206811|4947540.8546573|1100.5657959|1
364400.90887666|4947538.51907409|1100.55358887|1
364405.32568521|4947536.18349089|1100.66027832|1
364409.76115823|4947533.88553272|1100.69372559|1
364414.29306415|4947531.78197016|1100.84313965|1
364418.82497007|4947529.6784076|1100.84924316|1
364423.35687598|4947527.57484505|1100.83398438|1
364427.8887819|4947525.47128249|1100.98937988|1
364432.42068782|4947523.36771993|1101.05334473|1
364436.73075196|4947520.86962065|1101.02600098|1
364440.88273033|4947518.09037213|1101.12658691|1
364445.0347087|4947515.31112362|1101.1418457|1
364449.18668706|4947512.53187511|1101.10827637|1
364453.33866543|4947509.7526266|1101.16320801|1
364457.4906438|4947506.97337808|1101.17236328|1
364461.310806|4947503.78232435|1101.17541504|1
364464.90948861|4947500.31640022|1101.20581055|1
364468.50817123|4947496.85047609|1101.2911377|1
364472.10685385|4947493.38455196|1101.40698242|1
364475.70553647|4947489.91862782|1101.30944824|1
364479.13345717|4947486.29495697|1101.34301758|1
364482.30381197|4947482.43335146|1101.34301758|1
364485.47416677|4947478.57174595|1101.38867188|1
364488.64452157|4947474.71014044|1101.38867188|1
364491.81487636|4947470.84853492|1101.43444824|1
364494.98523116|4947466.98692941|1101.46496582|1
364497.7929403|4947462.86541442|1101.45275879|1
364500.43117195|4947458.62243428|1101.4588623|1
364503.06940359|4947454.37945414|1101.52587891|1
364505.70763523|4947450.136474|1101.53198242|1
364508.34586687|4947445.89349386|1101.57165527|1
364510.98409851|4947441.65051372|1101.62036133|1
364513.06486498|4947437.11870567|1101.64782715|1
364515.00948083|4947432.51635672|1101.6965332|1
364516.95409668|4947427.91400777|1101.74536133|1
364518.89871252|4947423.31165881|1101.70263672|1
364520.98137967|4947418.78620214|1101.74841309|1
364523.87058663|4947414.70997474|1101.81237793|1
364526.7597936|4947410.63374735|1101.83984375|1
364529.64900056|4947406.55751995|1101.88562012|1
364532.53820752|4947402.48129255|1101.95263672|1
```

'v.out.ascii input=Banks1_points_elev@PERMANENT' copied to clipboard

Figure 6. The list created from the attribute table of the point vector map with elevation values.

The rest of the viewing and analyzing will take place in the associated RStudio notebook, provided.

Summary

In this handout, we learned how to generate transects and convert lines into points, that we then used to extract values from a raster for further assessment. With this workflow, we are able to get the slope and sinuosity values needed for the Rosgen classification! The final workflow will walk you through how to use our dataset to classify your datapoint.