

User Manual for Ohio Wetland Classification and Channel Extraction

Software Used in This Research

Required

1. ESRI ArcGIS Desktop (<https://www.esri.com/en-us/arcgis/products/arcgis-desktop/overview>) or ArcGIS Pro (<https://www.esri.com/en-us/arcgis/products/arcgis-pro/resources>)

ArcGIS is the main tool and is used for all spatial data processing. Both the classic ArcGIS Desktop and the newer ArcGIS Pro work. ArcGIS is a commercial software.

2. MATLAB (<https://www.mathworks.com/products/matlab.html>)

MATLAB is used for image classification. There are many tools for image (data) classification, and some are available in open-source software. MATLAB provides some easy-to-implement toolboxes. Optionally, the user can install the Image Processing Toolbox (<https://www.mathworks.com/products/image.html>) and Deep Learning Toolbox (<https://www.mathworks.com/products/deep-learning.html>). MATLAB is a commercial software.

3. LAStools (<https://rapidlasso.com/lastools/>)

LAStools is a free & commercial software for lidar data processing. This research only used the free functions in this toolbox.

4. Microsoft Visual Studio Community (<https://visualstudio.microsoft.com/>)

Visual Studio is used for compiling C++ codes on Windows platform to process image & text data. The Community version is free.

Optional

5. Python (<https://www.python.org/>)

Though all ArcGIS processing steps can be done in the user graphic interface (ArcMap or ArcGIS Pro), the user is recommended to understand basic Python language syntax. This is because those ArcGIS tools can be equally implemented in ArcPy scripts, typically one line for one function. Here ArcPy is the ArcGIS toolbox for Python; it will be automatically installed with ArcGIS software.

6. Notepad++ (<https://notepad-plus-plus.org/downloads/>)

Notepad++ is used for viewing and editing text files, usually larger than what Windows Notepad can handle, up to 2GB. For files larger than 2GB, one may use the commercial software UltraEdit (<https://www.ultraedit.com/>).

Downloading Data

The aerial images and lidar point clouds can be downloaded from the Ohio Geographically Referenced Information Program (OGRIP) website at <https://gis5.oit.ohio.gov/geodatadownload/default.htm> (Figure 1). The Phase 1 provided a complete coverage of color infrared (CIR) images for the whole state. The Phase 2 was missing for some counties. The Phase 3 provided better coverage than Phase 2 in some watersheds. By 04/2022, only Phase 1 provided a statewide lidar point clouds, and future lidar collections are recommended when available.

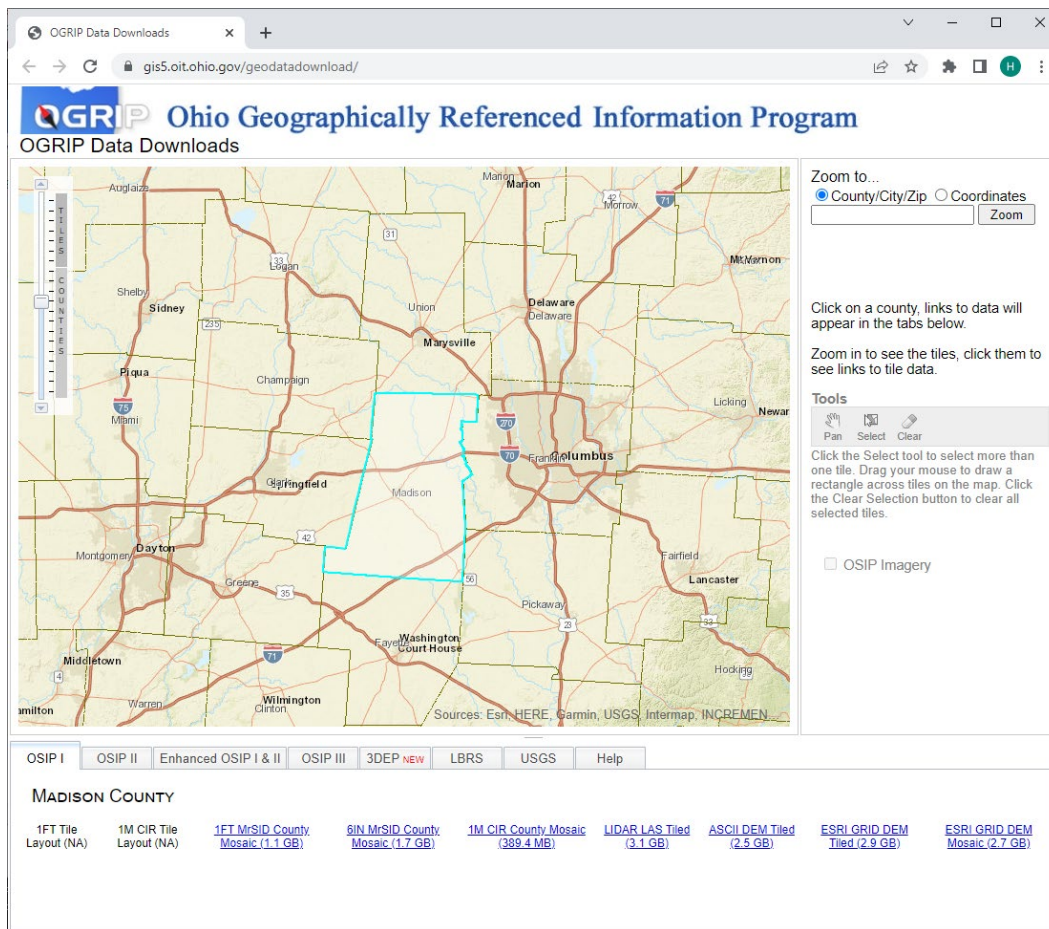


Figure 1. Downloading aerial images and lidar point clouds from the OGRIP website.

This research resampled every MrSID raster layer to 1 m resolution. Some raw data may provide 6-inch (15 cm) resolution but processing such high-resolution data requires substantially more computing resources and therefore is not recommended.

After downloading the images and lidar, unzip the compressed files and save them to the hard drives.

Tips: A typical file format for those aerial images is *MrSID* with the file name extension *.sid*. The *MrSID* is a compressed file format the converted (resampled, reprojected) images are usually larger than the *MrSID* files. If you use TIF to store images, enable TIF compression to save space. The ESRI file geodatabase raster layers are also compressed. Please **DO NOT** store file geodatabases on a cloud drive (e.g., OneDrive). Synchronizing a file geodatabase between the local file system and the cloud server,

and between different computers may cause file corruptions and the raster layers may be permanently lost, though you may recover some vector layers by the Recover File Geodatabase tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/data-management-toolbox/recover-file-geodatabase.htm>).

Preparing Image Layers

The OGRIP data are organized by county and the user usually needs to mosaic several counties to cover the study site. First, reproject and resample the images to the desired resolution. For local study with high spatial resolution, the State Plane Coordinate System (SPCS) is recommended due to its high precision. Ohio has two SPCS projection, *Ohio North* and *Ohio South* (<https://geodesy.noaa.gov/SPCS/maps.shtml>). For a study site near the N-S boundary, the user should decide which projection to use and project some images to the desired projection (Figure 2). Meanwhile, the image resolution should be resampled to the desired value. The ArcGIS Project Raster tool is used for the above two purposes (Figure 3).

(<https://desktop.arcgis.com/en/arcmap/latest/tools/data-management-toolbox/project-raster.htm>)

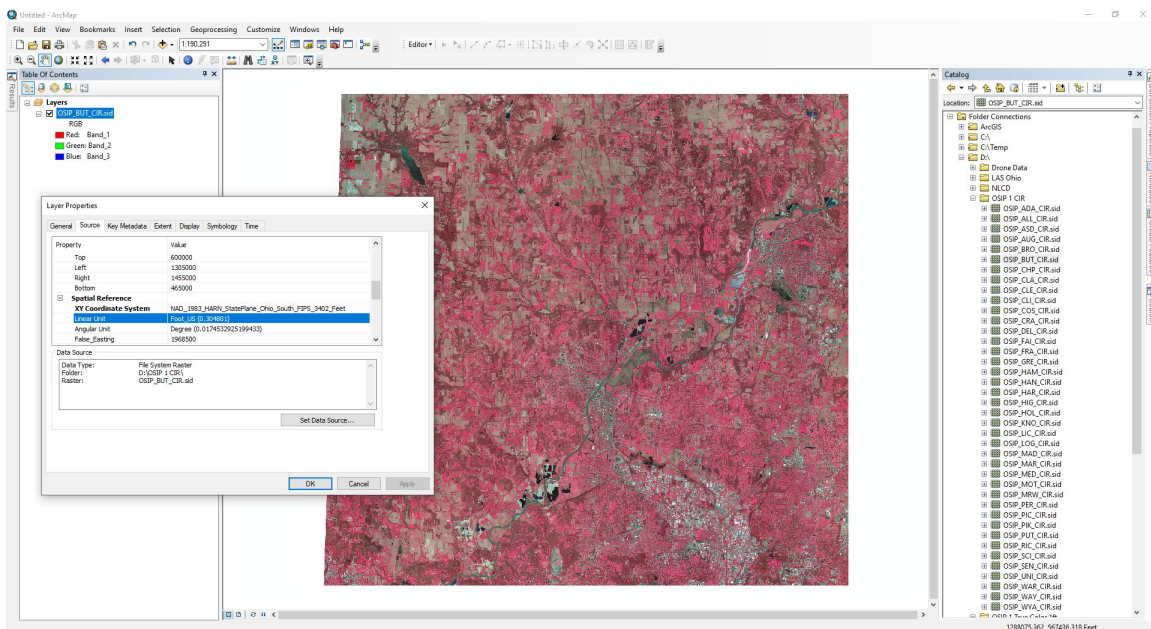


Figure 2. Checking the image properties. Here shows the CIR image of Butler County, OH. The raw image is in SPCS Ohio South and has the resolution of 3 ft.

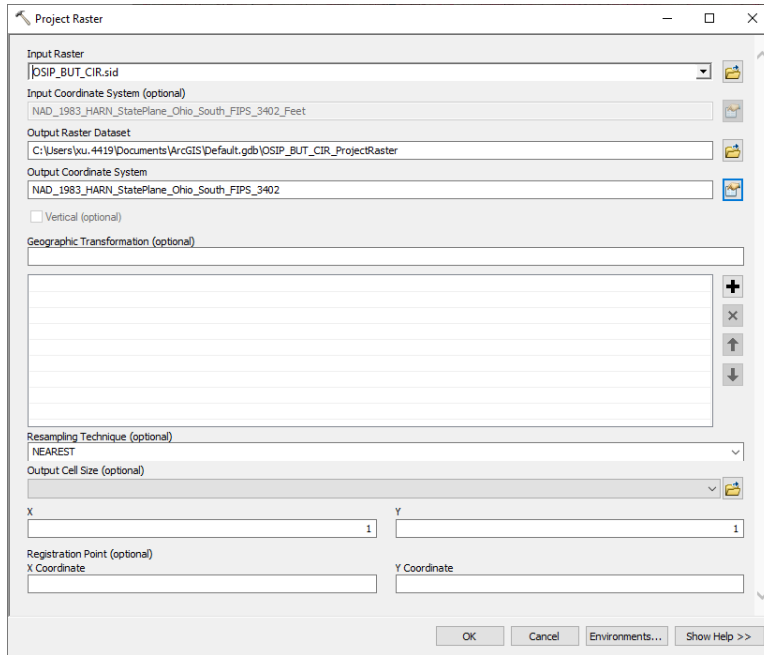


Figure 3. The Project Raster Toolbox. The user should specify the output projection and the cell size. Remember, this tool does two jobs: reprojecting and resampling.

After the projection, check the image property again to make sure the output raster is correct (Figure 4).

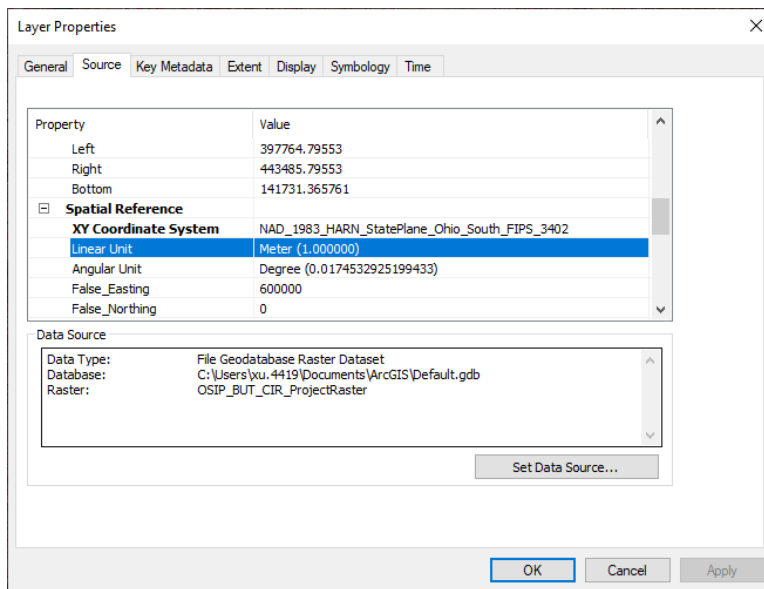


Figure 4. Image properties after projection and resampling. Here the projection is *State Plane Ohio South* and the resolution is 1 m.

The next step is clipping the images (Figure 5). This is necessary because some images have white areas near the county boundary; the white areas will cause problems when mosaicking images. Use a county boundary layer (<https://www.census.gov/geographies/mapping-files/time-series/geo/carto-boundary->

[file.html](#)) for this step. Select the county polygon before running the Clip Raster tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/data-management-toolbox/clip.htm>)

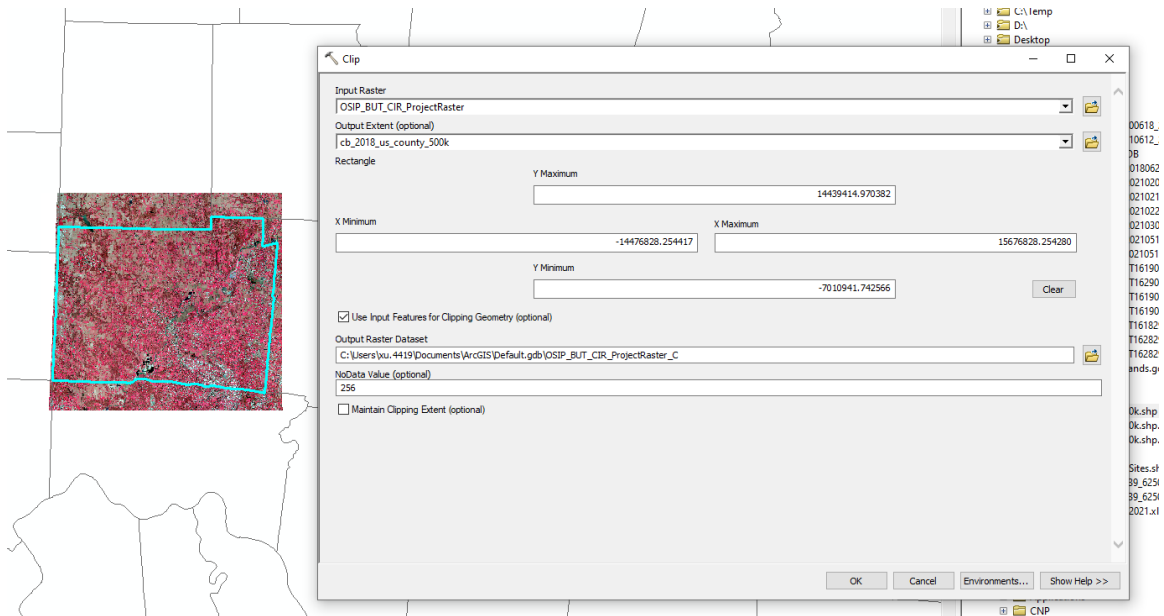


Figure 5. Clipping the image. Note Butler County polygon has been selected (highlighted in cyan color) and the *Use Input Features for Clipping Geometry* checkbox is checked.

Tips: You must select the polygon (highlighted) unless the vector layer has only one polygon feature, or the clipping boundary will be different. After this step, you may delete the reprojected image as the result of Figure 3, because that image will never be used after this step.

For most HUC8 and smaller study sites, one can use a new raster dataset (Figure 6) for the mosaic image. First, create an empty raster dataset in the Catalog dialog (Figure 6). This must be created under a file geodatabase (<https://desktop.arcgis.com/en/arcmap/latest/manage-data/geodatabases/types-of-geodatabases.htm>).

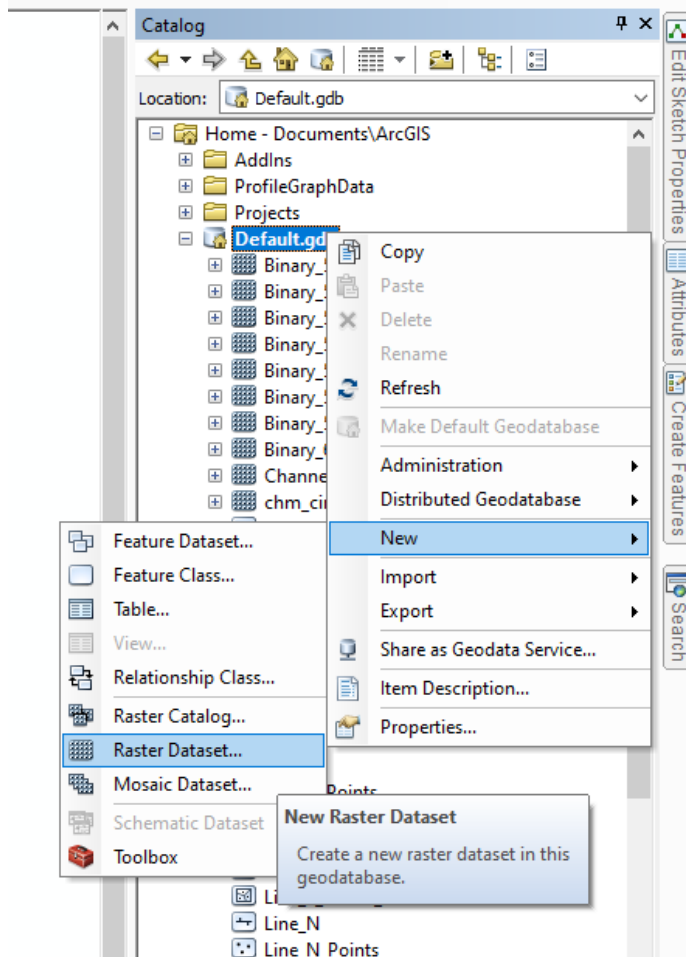


Figure 6. Creating a raster layer for the mosaic.

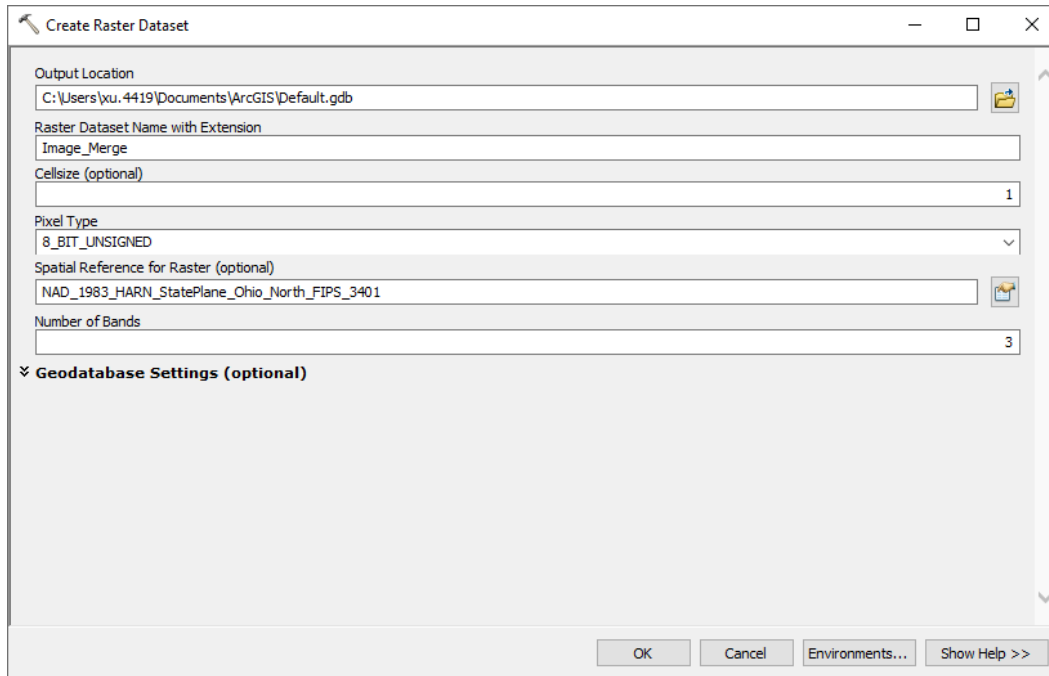


Figure 7. Parameters for creating the empty raster layer.

The typical *Pixel Type* of an aerial image is 8-bit unsigned integer, sometimes written as UINT8. The UINT8 range is in $[0, 255]$ with 256 possible values. Here 255 equals 2^8-1 . Therefore, the pixel type should be consistent between the images (Figure 7). Also, specify the map projection and the number of bands. A typical aerial image has three bands. Sometimes the data provider claims four bands for color infrared aerial images; the user should verify this and decide the correct number of bands.

Tip: You do not need the Mosaic Dataset unless you are working on a large dataset, e.g., half of the State of Ohio, or a study area with hundreds of image tiles.

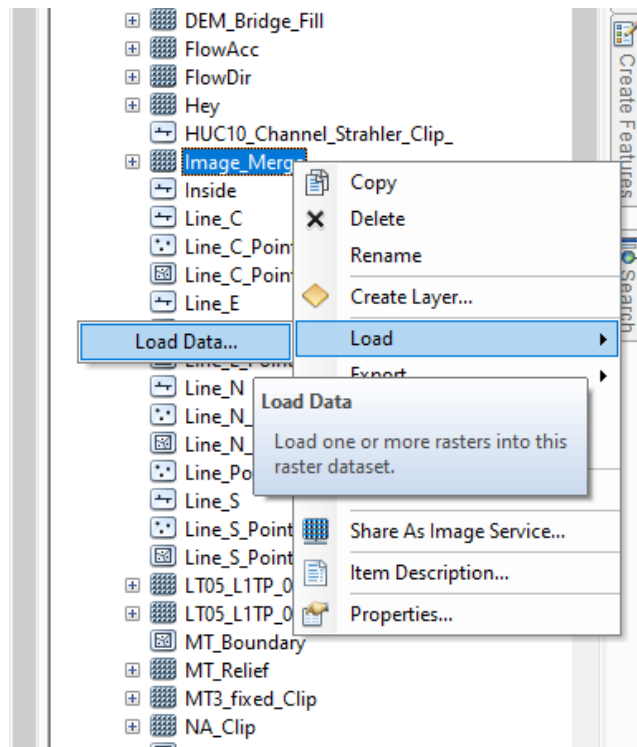


Figure 8. Loading images to the newly created empty image.

The next step is to load the clipped images to this empty raster layer (Figure 8). In the toolbox dialog (Figure 9), choose all the layers for the mosaic and leave all other parameters as default.

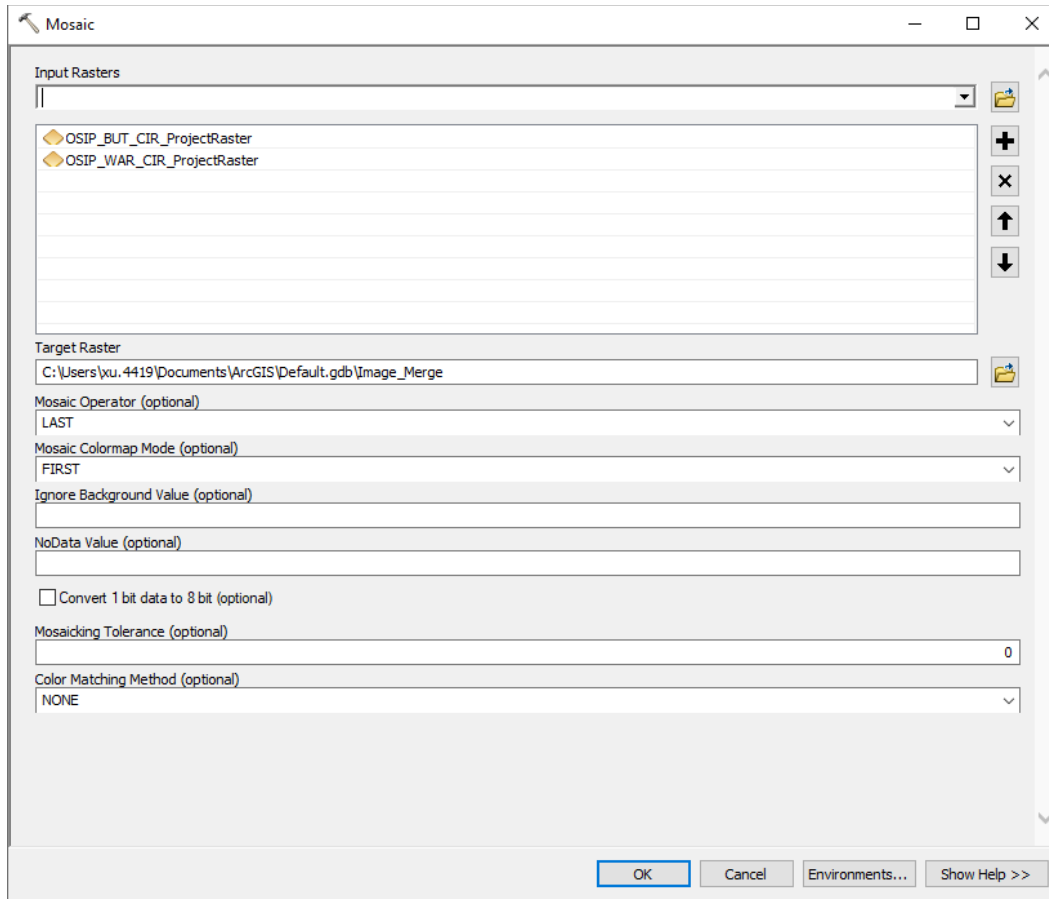


Figure 9. Loading images to the newly created empty image.

Tip: The Mosaic Tool may take a long time. Check the progress at Geoprocessing->Results. You will see which image layers have been loaded.

The final step is clipping the mosaicked image (Figure 10). You need a polygon boundary layer which defines your study site. This research used watershed boundaries, found in the National Hydrography Dataset (NHD, <https://apps.nationalmap.gov/downloader/>). Download the NHD geodatabase for Ohio and unzip it. Find the Watershed Boundary Dataset (WBD) layers which are the polygon layers at different scales. The HUC8 is the largest unit tried in this research, and the user is recommended to use a smaller unit, such as HUC10 or HUC12.

It is recommended to slightly buffer the watershed boundary by 20 to 100 m, because 1) the boundary may not be precise enough for the high-resolution images; 2) the boundary may be inaccurate for terrain analysis on high resolution topography; and 3) for deep learning classification, a buffer around the central pixel is required.

Use the Clip raster tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/data-management-toolbox/clip.htm>) to clip the mosaicked images. This is the same tool used for clipping each county's image.

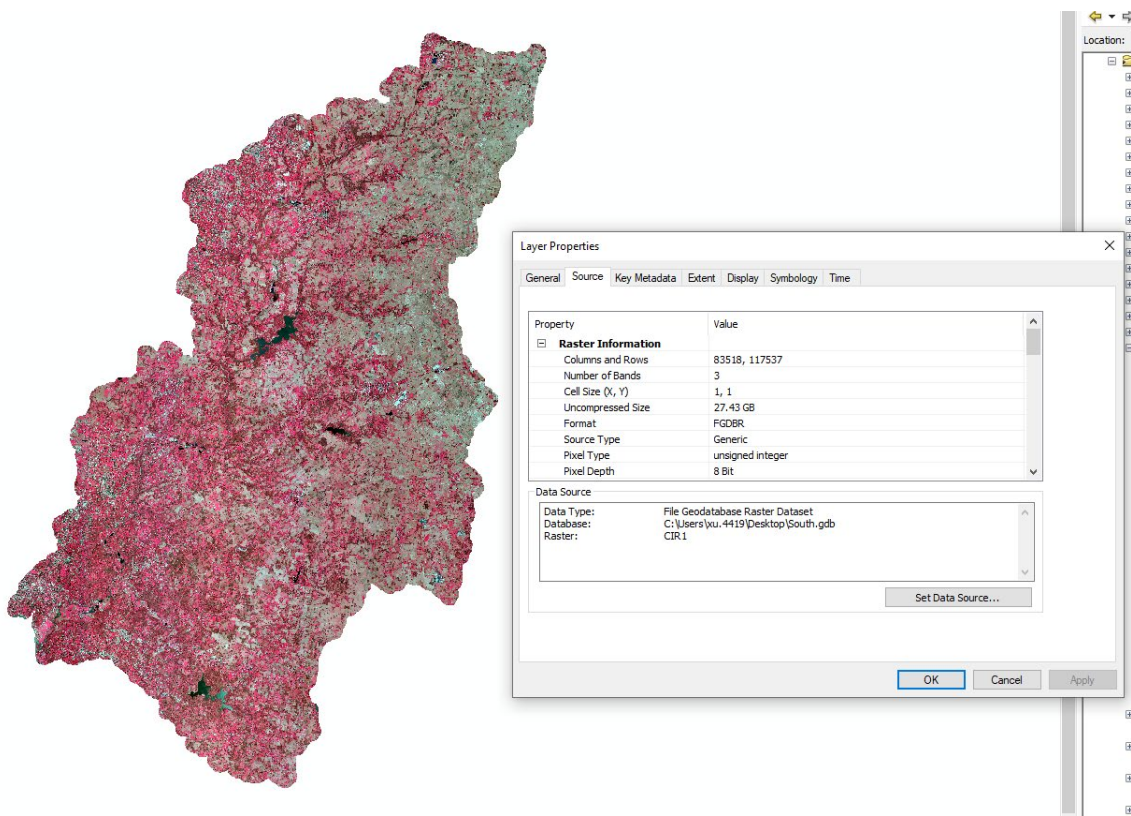


Figure 10. The final mosaicked and clipped CIR image for the Little Miami HUC8 watershed. Note the large size of the image even using UINT8 data type. UINT8 is typically the data type using the least amount of space. If using floating number (32 bit), the file size will be four times of UINT8.

Calculating NDVI

The normalized difference vegetation index (NDVI) is a common feature for differentiating ground features. NDVI is a ratio involving the near-infrared (NIR) band and the red band, defined as below.

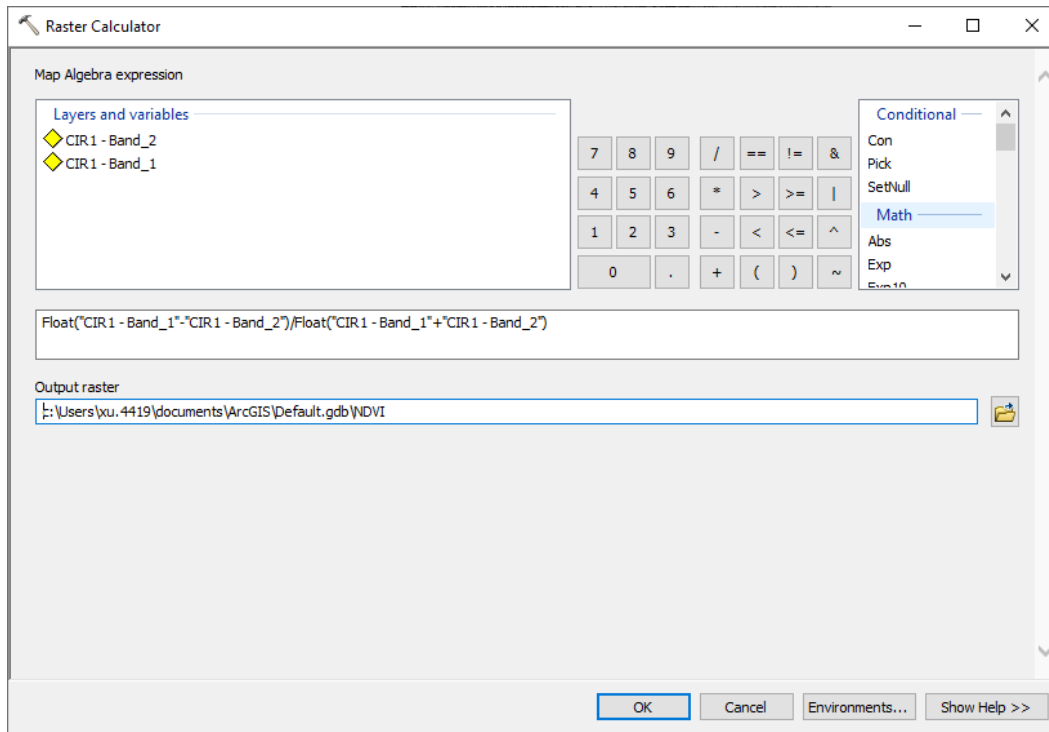
$$NDVI = \frac{NIR - Red}{NIR + Red}$$

The Phase 1 CIR images has three bands: the first two being NIR and red, respectively. Therefore, the NDVI is calculated in the Raster Calculator tool in Figure 11

(<https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/raster-calculator.htm>).

Note there are two bands in Figure 11, this is because they were loaded in ArcMap separately. Do not load the entire multi-band image, instead, unfold the bands, and load the required bands.

Note the Float() function is enforced because the default data type is associated with the input image (UINT8). If Float() was ignored, then the output layer has only two values: 0 and 1. The correct output raster layer is a floating number layer (32-bit) with the range [-1, 1]. The user may check the zero-denominator issue (NIR + Red = 0), but it rarely occurs.



Now we must decide the data type. The NDVI is in 32-bit float, and all other image layers except NDVI are in UINT8. If we convert UINT8 to float, we don't lose any information, but we don't gain any new information. Meanwhile, the storage size will be four times as before. If we convert 32-bit float to UINT8, we lose some information but keep the storage size low. This research used the latter solution.

We will linearly stretch the NDVI range [-1, 1] to UINT8 range [0, 255] (Figure 12). The equation is

$$\text{Int} ((NDVI + 1) * 127.5)$$

Here the Int() function converts a number to an integer. Given the range of NDVI, the converting result will be integers in the range of [0, 255]. Now check the data type of the converted NDVI. If it is a long int type (e.g., UINT16 or UINT32), then convert it to UINT8 using the Copy Raster tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/data-management-toolbox/copy-raster.htm>).

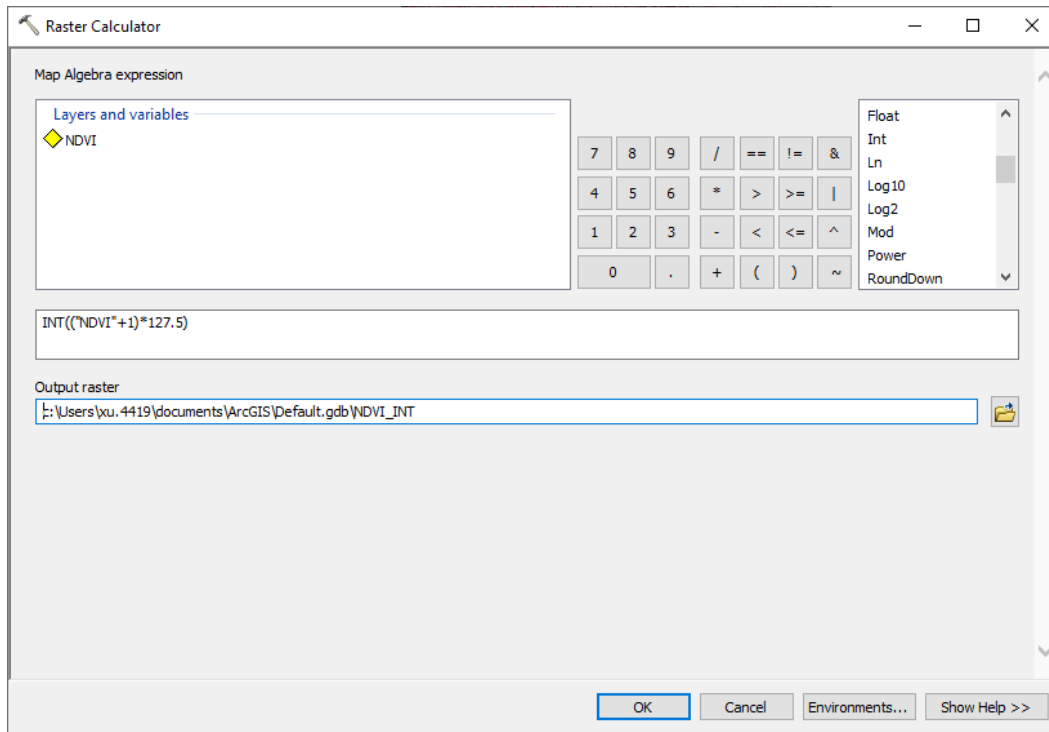


Figure 12. Converting NDVI floating numbers to UINT8 integers.

Preparing Lidar Layers

Both topographic and vegetation information will be extracted from lidar data. But they are usually not directly provided, and we will extract them from lidar point clouds.

Contemporary lidar point clouds in the US are usually provided in a zipped format, i.e., LAZ file format. ArcGIS does not directly read this format but reads the unzipped format, LAS. To unzip LAS, one can use the free tool provided by the LAStools (Figure 13). Download LAStools and unzip it. Find the *laszip.exe* tool under the *bin* folder.



Figure 13. The LAsTools software for lidar data processing.

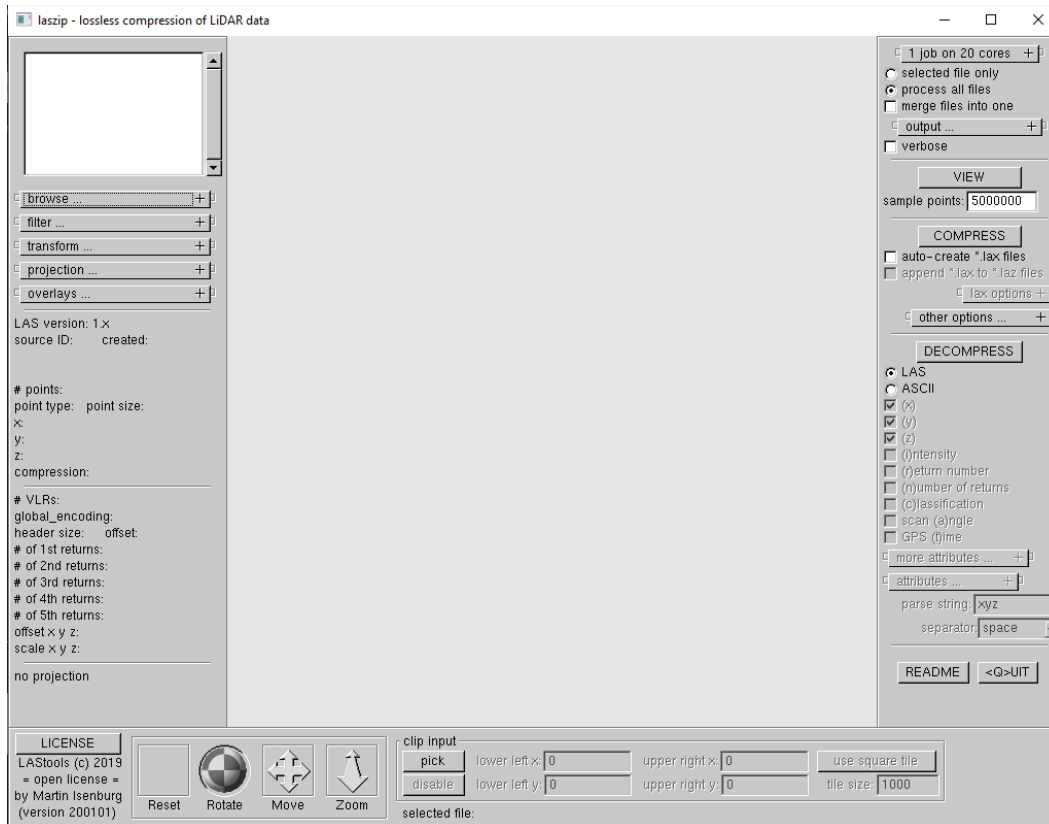


Figure 14. The *laszip.exe* tool from LASTools.

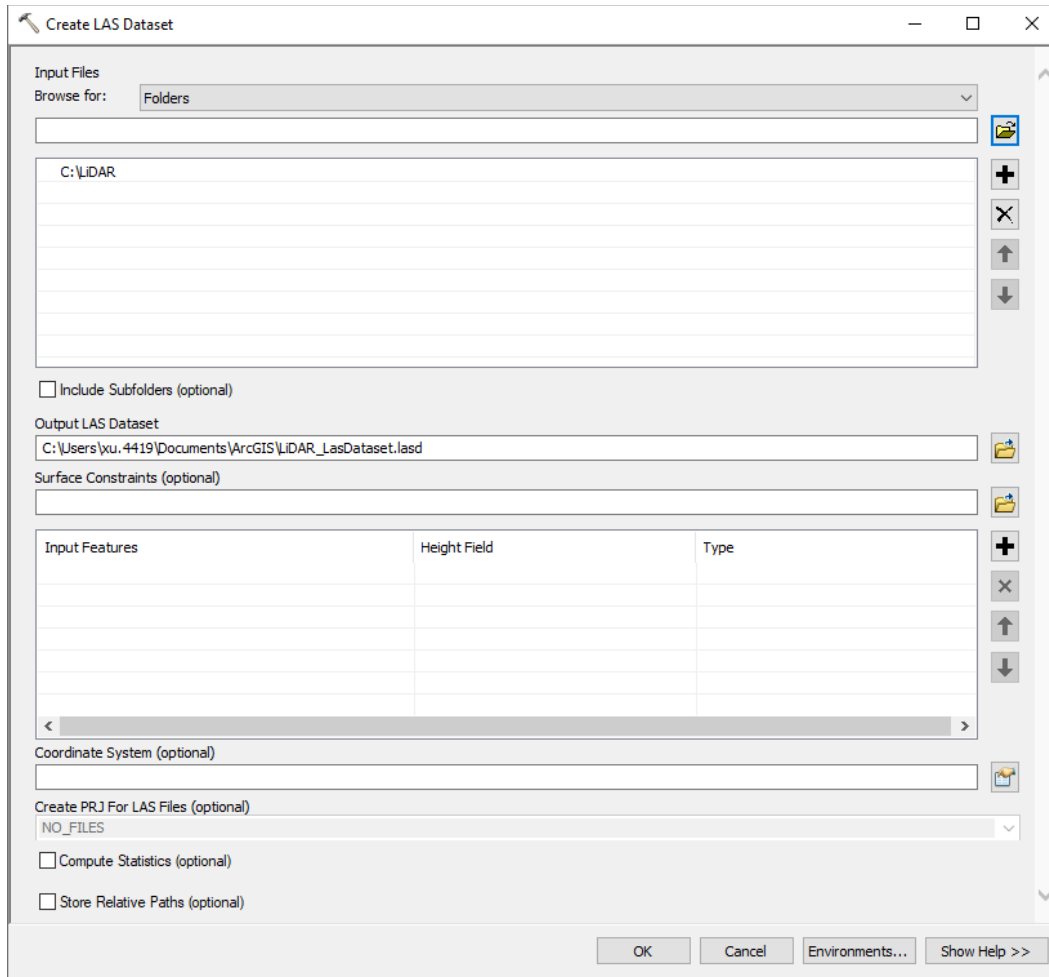


Figure 15. Creating a las dataset.

<https://desktop.arcgis.com/en/arcmap/latest/manage-data/las-dataset/creating-a-las-dataset.htm>) and select the folder containing all LAS tiles (Figure 15). Specify the output file (*.lasd).

Tip: For the input file type, it is recommended to use Folders instead Files. Selecting individual LAS files is also possible, but when the number is large this may take a long time.

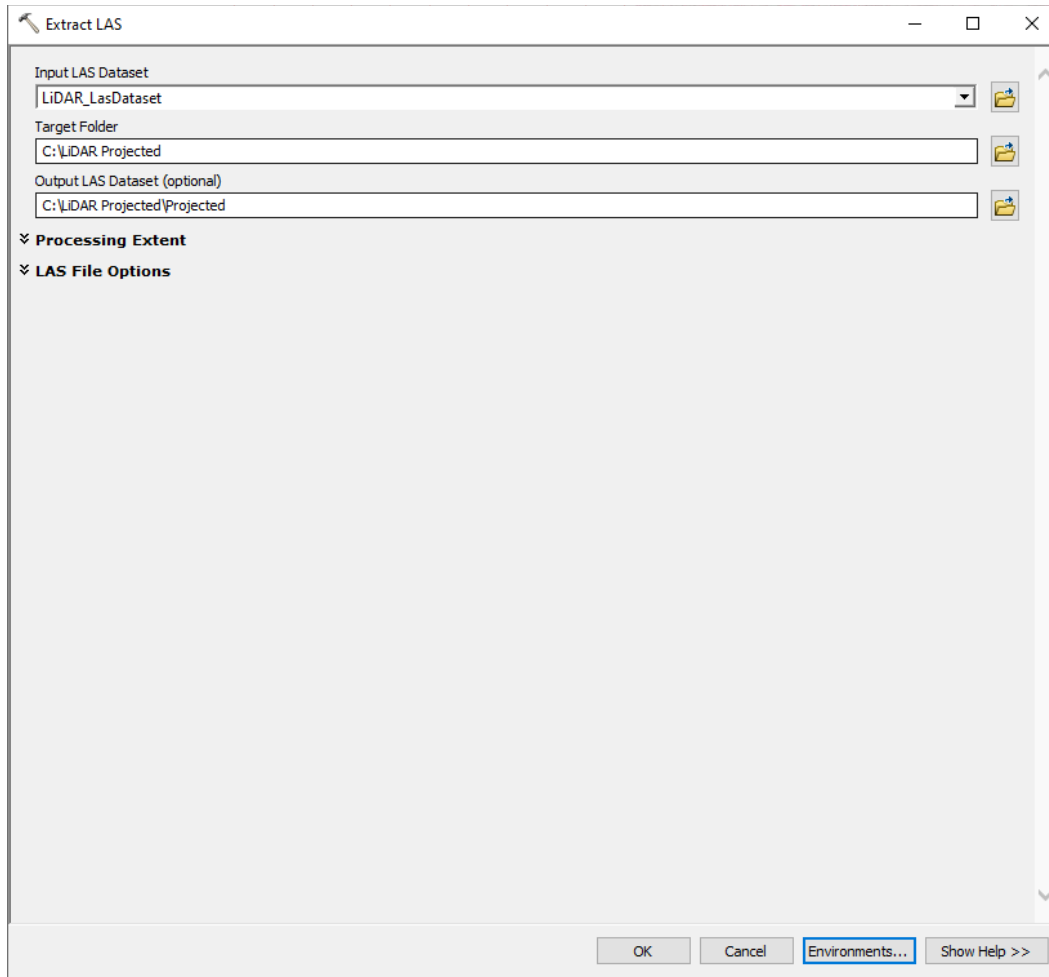


Figure 16. Projecting LAS tiles.

<https://desktop.arcgis.com/en/arcmap/latest/tools/3d-analyst-toolbox/extract-las.htm>) and select the las dataset in the input. Then specify the output direction, where the projected LAS tiles will be saved. Click the **Environments** button near the bottom of the dialog and specify the **Output Coordinates** (Figure 17).

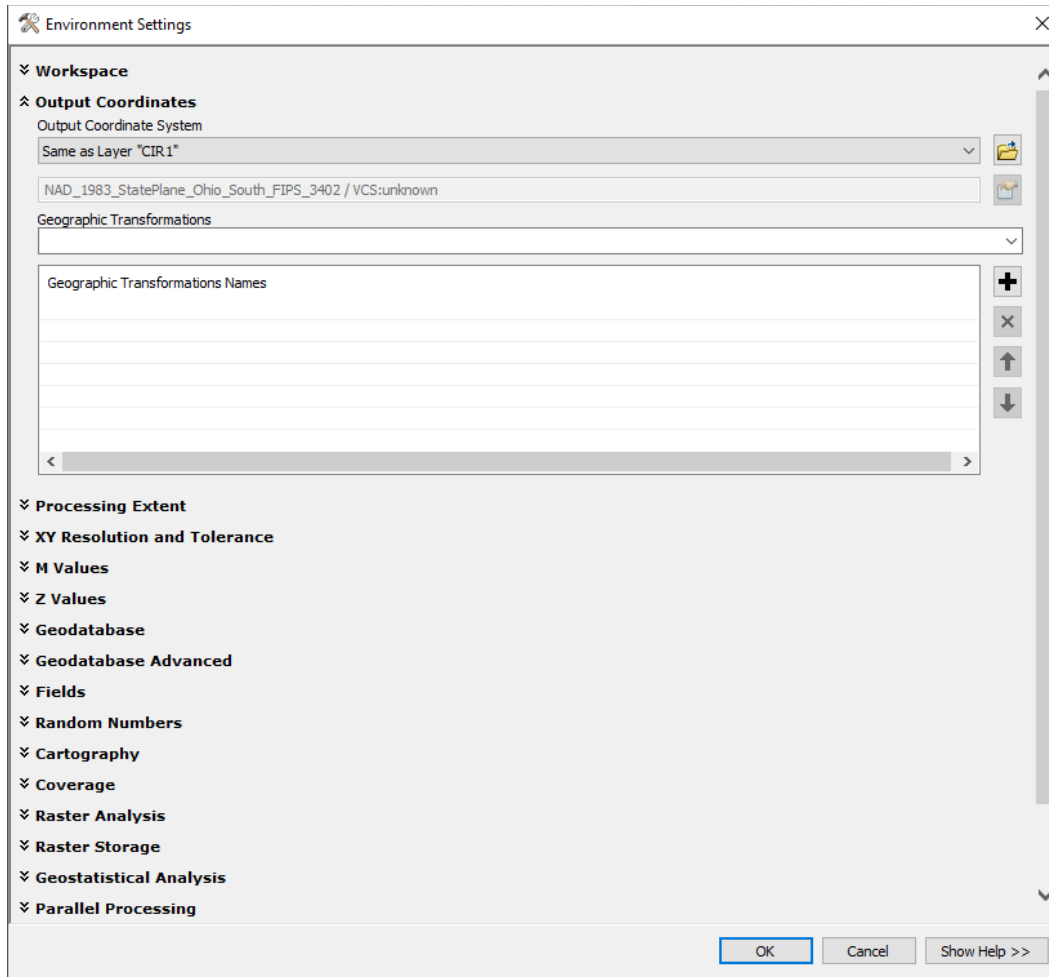


Figure 17. Setting the output coordinates.

In ArcMap, right click the LAS Dataset layer name and open the Layer Properties dialog (Figure 18). Click **Ground** button so only ground returns will be displayed and used. Use the LAS Point Statistics As Raster tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/data-management-toolbox/las-point-statistics-as-raster.htm>), and select the las dataset with the correct ground filters (Figure 19). Choose **POINT_COUNT** for **method**. Pay close attention because the default value **PULSE_COUNT** looks very similar. For the output cell size, leave it as the same as images (1 m). Normally this value should be four or five times of the lidar point spacing, but we will use filters to smooth the output.

The output raster layer is an integer layer ranging from 0 to a positive number. This is the ground return *point per cell* and we use Reclassify (<https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/reclassify.htm>) to convert it to lidar void. In the Reclassify tool, let 0 become **1**, and let any positive number (e.g., 1 to 99999999) become **NoData**. The output layer is a binary raster with 1 being lidar void.

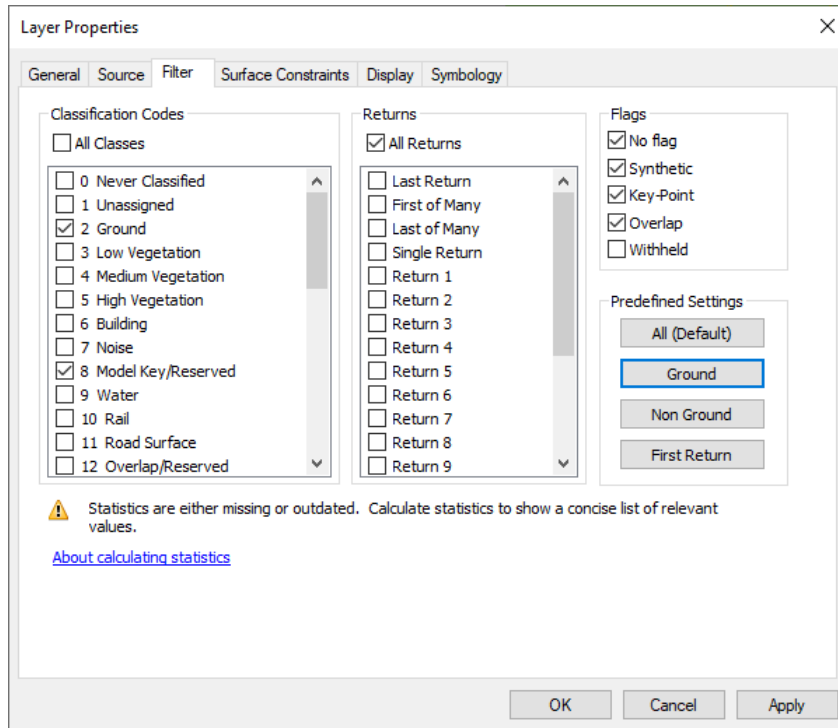


Figure 18. Filtering the lidar returns.

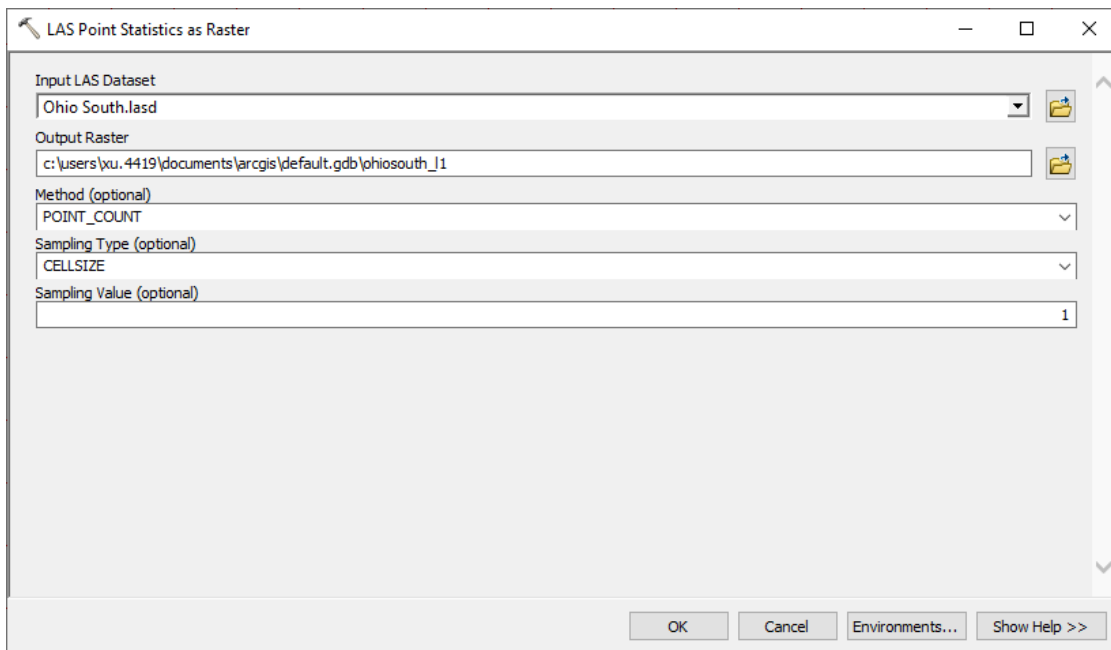


Figure 19. Creating the ground return density layer.

We do not include the lidar void layer in classification, but the layer provides important information and helps improve the canopy height layer. Topographic lidar systems shoot a laser beam at near infrared wavelength, which is strongly absorbed by water. Therefore, a lidar void implies surface inundation, whether it is vegetated or not. Those locations are usually open water bodies and inundated areas during the lidar mission. For the latter, they may be shallow depressions and thus are more likely to be wet.

Tip: Buildings also have no ground return and cause confusion. You can remove buildings from the lidar void layer with a land cover map such as National Land Cover Database (<https://www.usgs.gov/centers/eros/science/national-land-cover-database>).

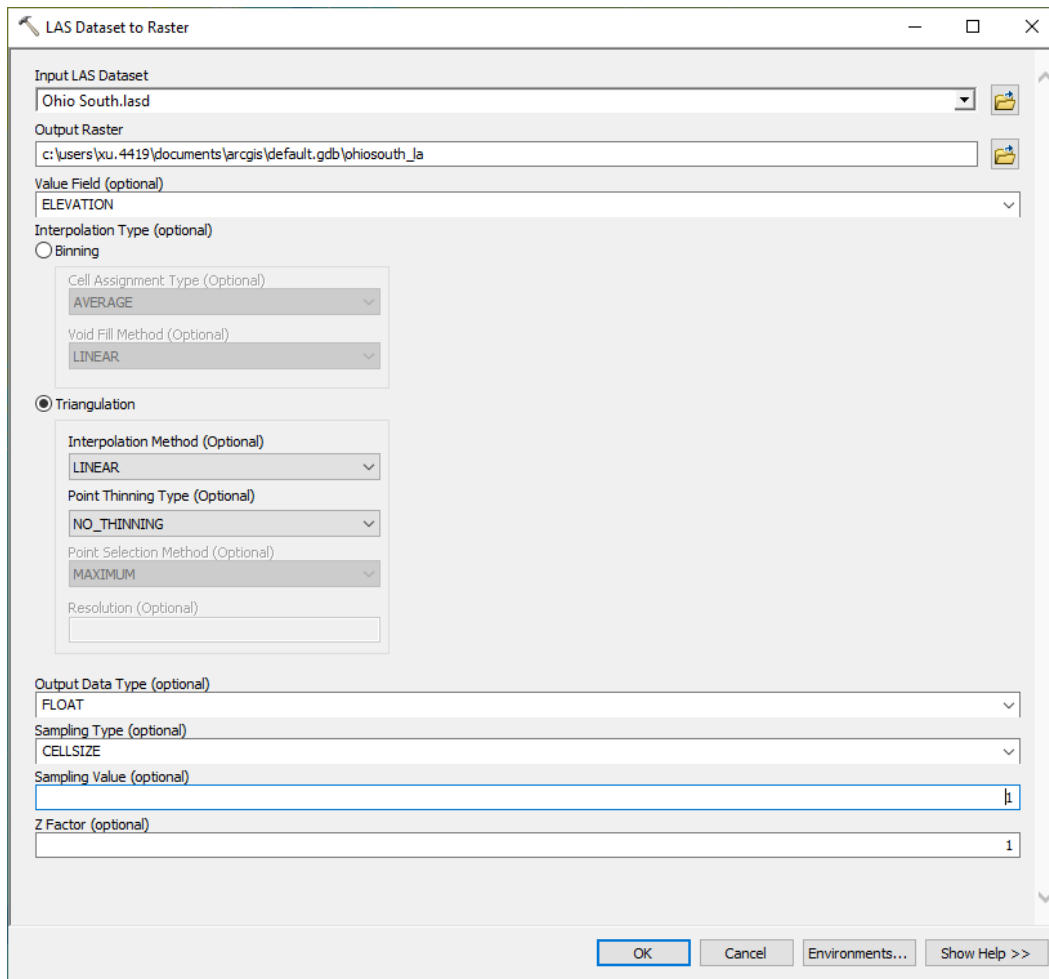


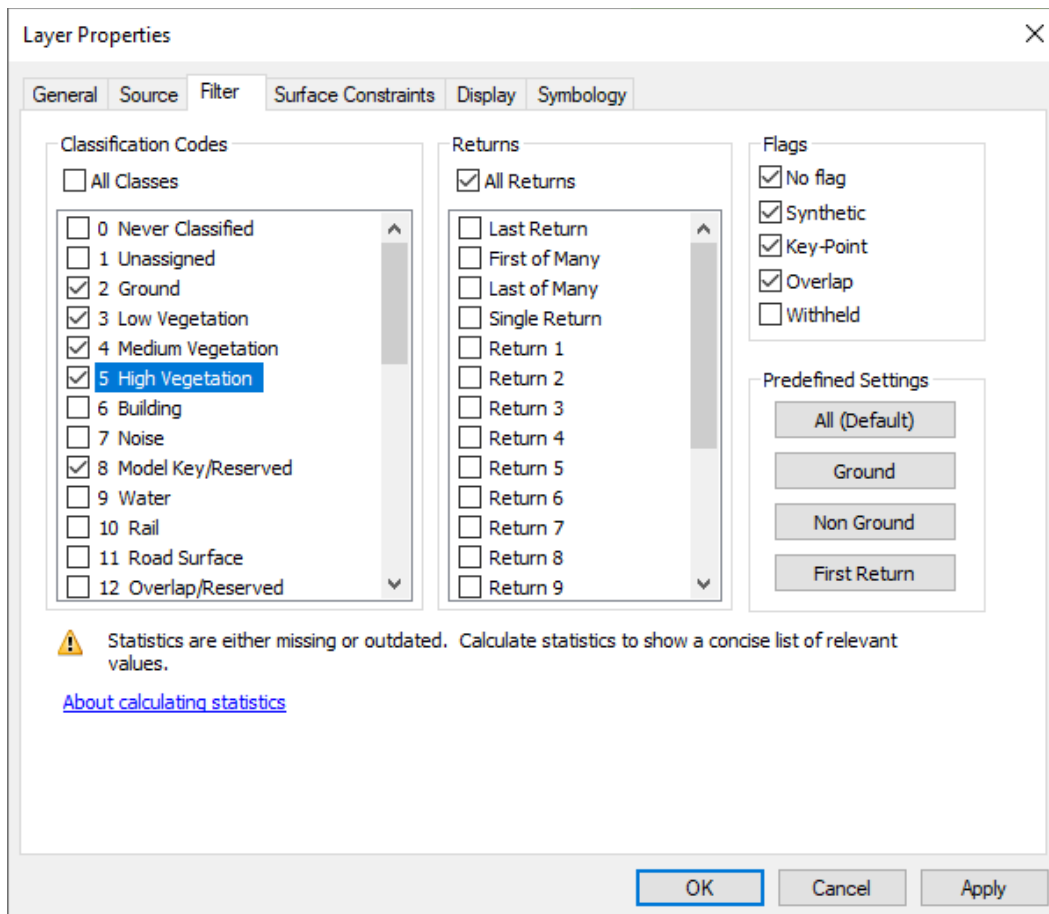
Figure 20. Creating DEM from the lidar point clouds.

<https://desktop.arcgis.com/en/arcmap/latest/manage-data/raster-and-images/las-dataset-to-raster-function.htm>). Choose **Triangulation** interpolation method and choose 1 (meter) for the cell size (Figure 20). The output layer will be the DEM. Then verify the DEM vertical unit is foot or meter. For channel extraction, the vertical unit does not matter. But for canopy height model

(CHM), you must know the unit (but not change it at this step). A quick tool to verify elevation is Google Earth (<https://earth.google.com/web/>).

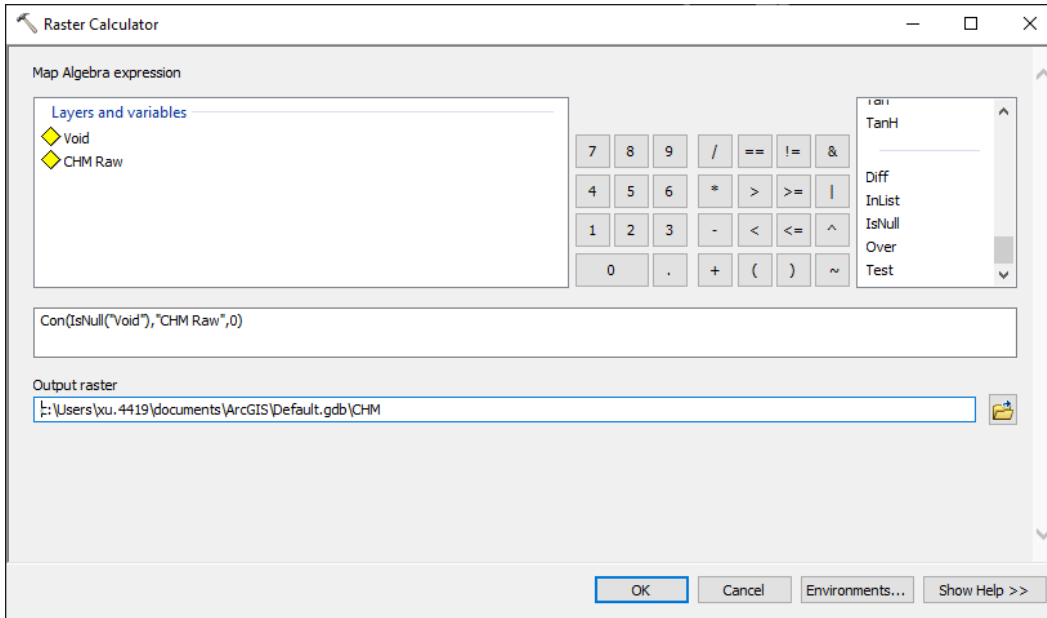
Tip: Specify the **Processing Extent** if your study site only covers part of the LAS Dataset. This will save you time creating the DEM. You can also change the output cell size to your desired value.

To create DSM, select low, medium, and high vegetation returns in addition to the ground returns (Figure 21). Then, use the LAS Dataset to Raster tool to create the DSM.

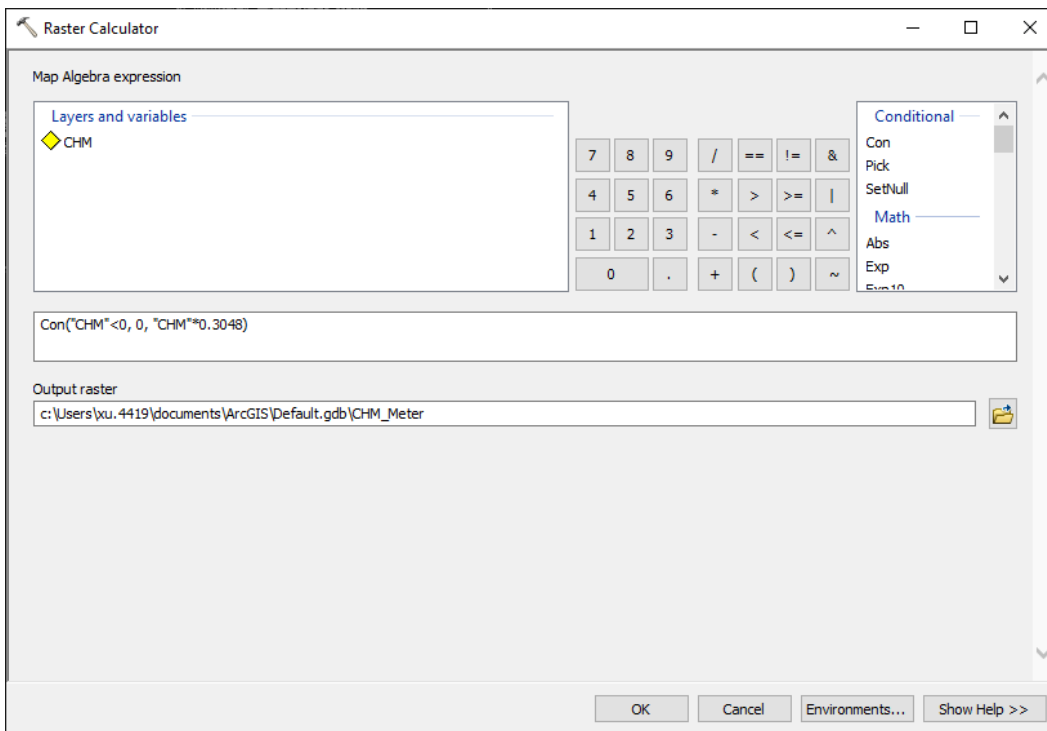


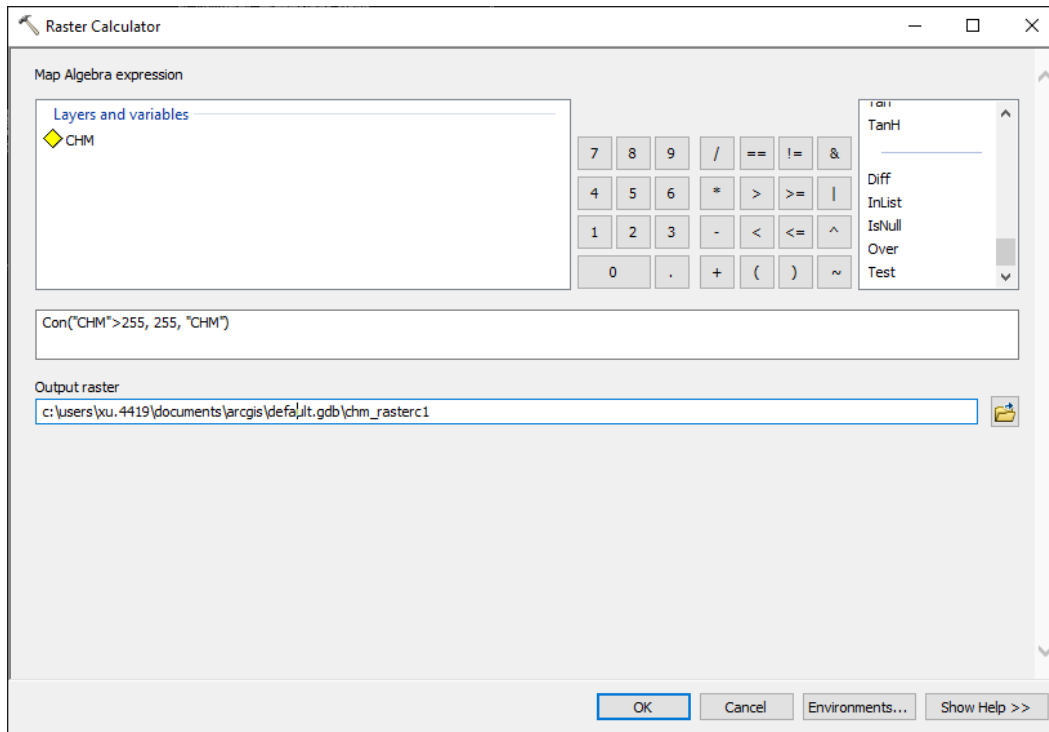
The DSM should have the same vertical unit as the DEM (remember the unit but no need to convert it now). The CHM is then calculated as the difference between DSM and DEM, using the Raster Calculator tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/raster-calculator.htm>).

The raw CHM usually contains errors of different sources. First, if you zoom in to any river, you may see some large triangles in the river, which are caused by no ground return in river and the triangulation method. Use the lidar void created above, set all CHM pixels to zero if it is also a lidar void (Figure 22).



Next, you may find the lowest value of the CHM layer is negative. Simply set any negative CHM to zero (Figure 23). Optionally, you may convert the vertical unit. It is required only if trees can grow over 255 ft, exceeding the UINT8 upper limit. In Ohio, the tallest trees are usually below 30 m or 90 ft (<https://www.monumentaltrees.com/en/records/usa/ohio/>). Therefore, you can use either ft or meter for CHM.

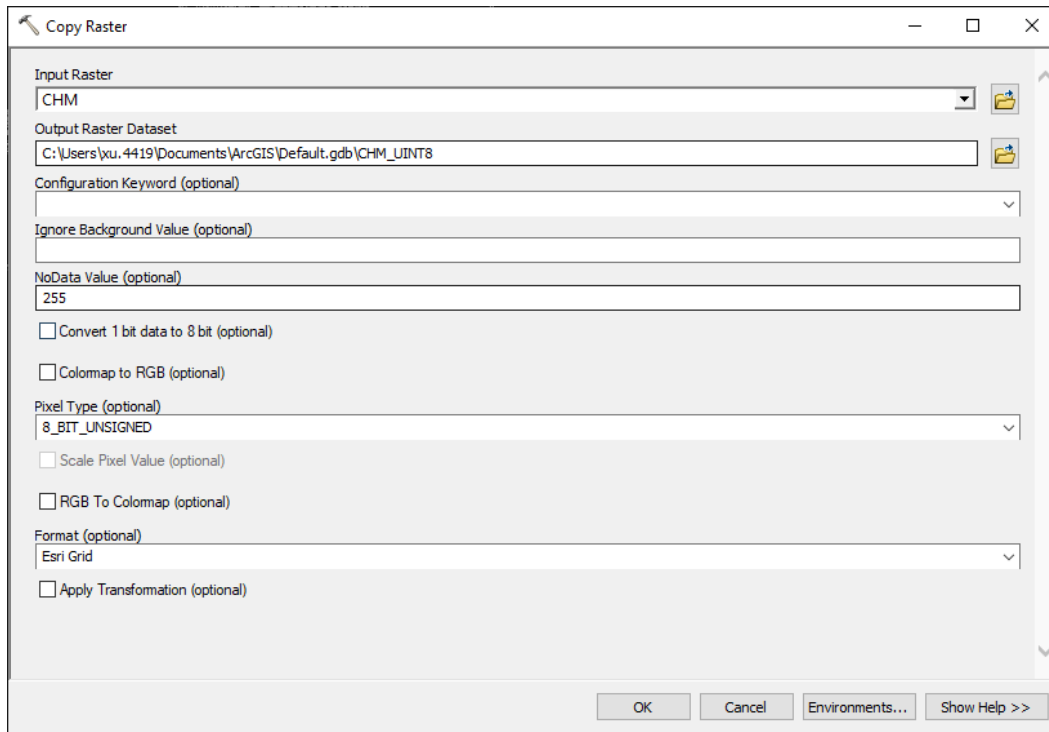




Since no tree in Ohio can grow to 255 ft or 255 m, any values over 255 are incorrect. Those extremely large values are usually caused by poles, power lines, birds, or errors in lidar. The easiest solution is to set all “trees” taller than 255 (ft or m) to 255 (ft or m), the upper limit of UINT8 (Figure 24).

Tip: You can lower the upper limit of the tree height, e.g., 50 m, because no tree in Ohio is above 50 m. Thus, you can write `Con("CHM">50, 50, "CHM")` in the code box.

Finally, the CHM is still in float data type, so we need to convert it to UINT8 using the Copy Raster tool (Figure 25). This conversion is not lossless but reduces the storage size by 3/4, like the NDVI conversion above.

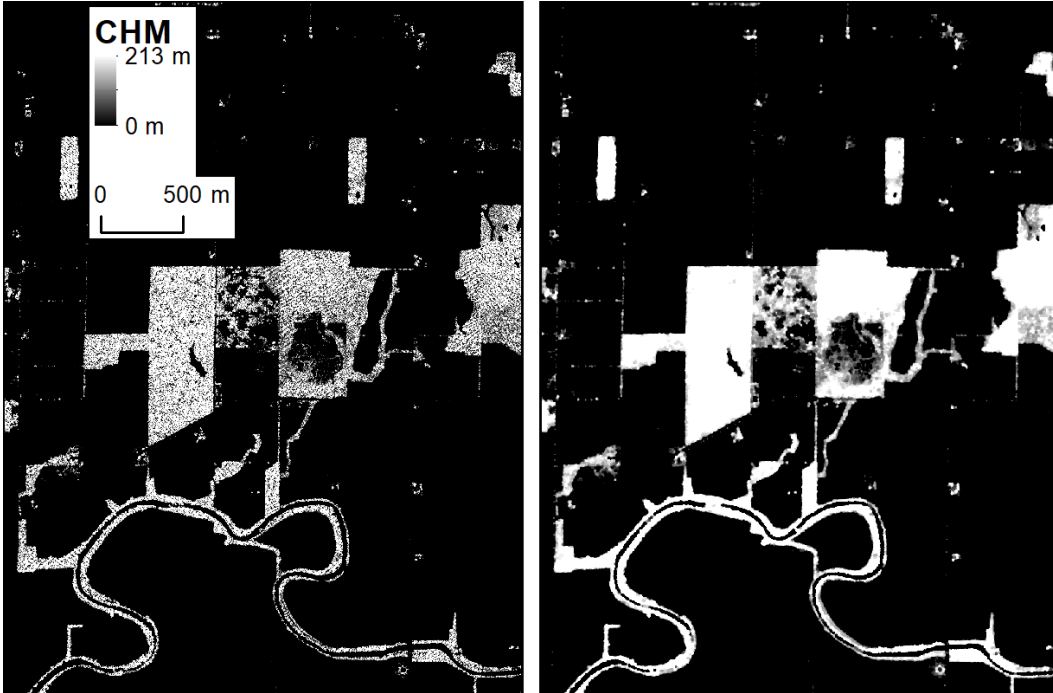


Morphological Filtering

The CHM still has noise and should be filtered before the image classification (Figure 26). Many image filters can be used. Here we choose the morphological filters to fill small gaps and remove small features (<https://desktop.arcgis.com/en/arcmap/latest/extensions/arcscan/using-raster-cleanup-morphological-operations.htm>). Note that we will use a different implementation described below.

The raw CHM contains the low value pixels (gaps) and should be filled (Figure 26). The exact steps are dilation (radius = 10 m), erosion (radius = 10 m), and dilation (radius = 3 m). Dilation equals the focal maximum operator, and erosion equals the focal minimum (Figure 27). Use Focal Statistics (<https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/focal-statistics.htm>) for those steps. The combination of dilation and erosion is also known as closing.

Furthermore: Morphological image processing gives intuitive results similar to the interpretation by human eyes. It works on both binary and grayscale images. The processing in Figure 26 is based on grayscale because the value range is [0, 255].



Focal Statistics

Input raster
CHM

Output raster
C:\Users\xu.4419\Documents\ArcGIS\Default.gdb\CHM_Dilation_R10m

Neighborhood (optional)
Circle

Neighborhood Settings
Radius: 10
Units: Cell Map

Statistics type (optional)
MAXIMUM

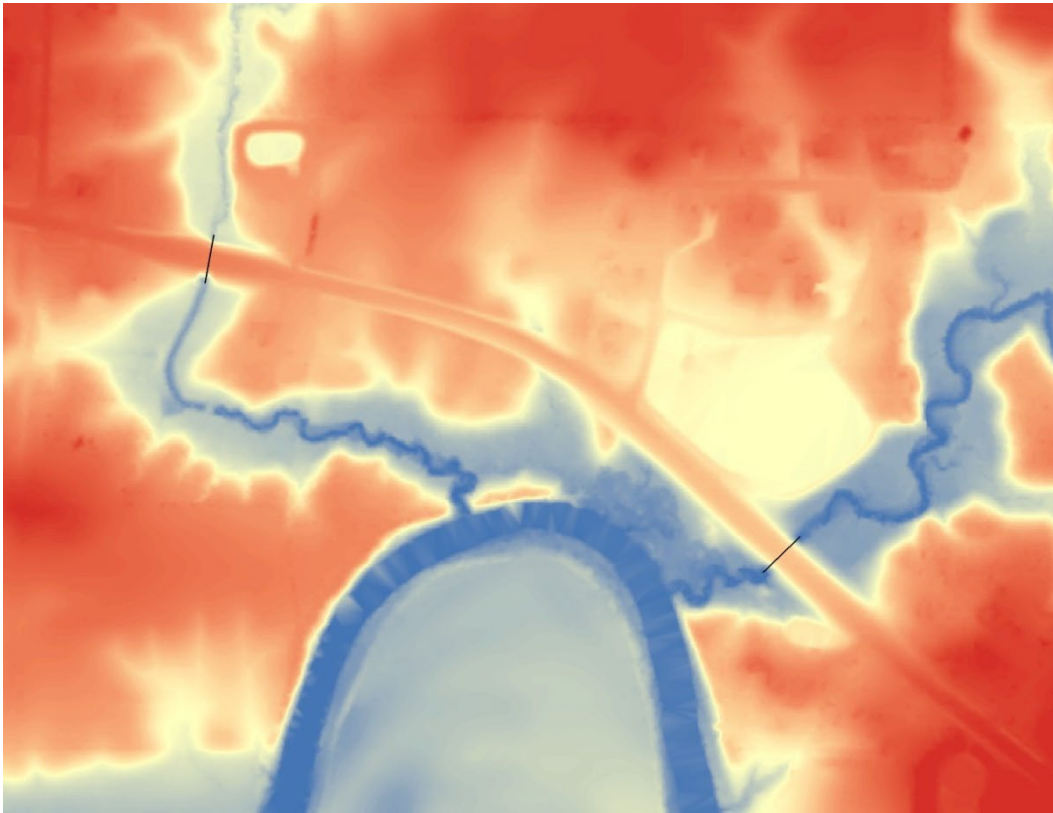
Percentile value (optional)
90

Ignore NoData in calculations (optional)

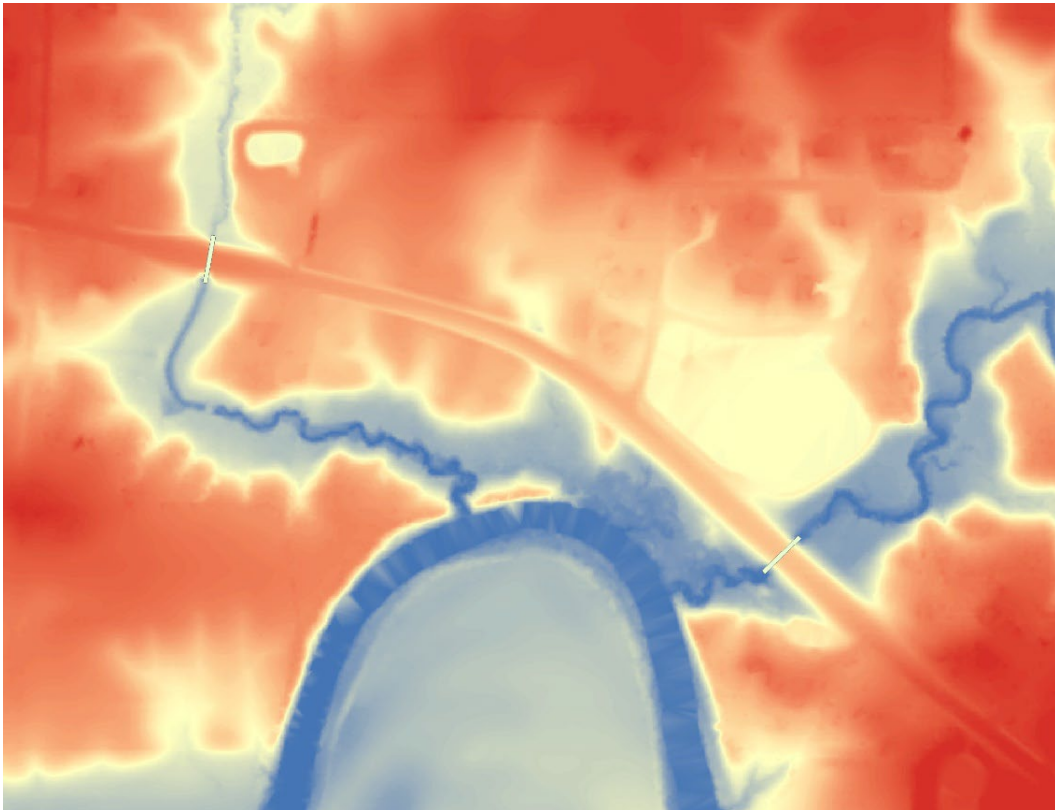
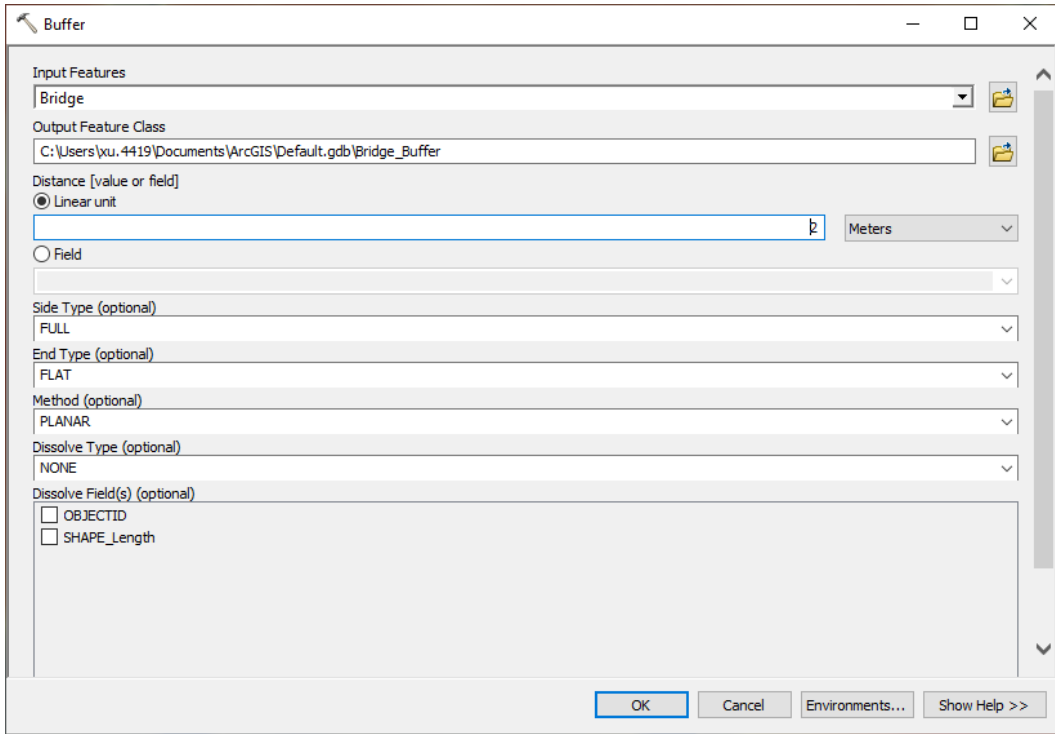
OK Cancel Environments... Show Help >>

Hydrologic Enforcement

Channel extraction is based only on lidar DEM. The raw DEM needs some processing to reduce errors in the extraction. Hydrologic enforcement is the first step (Figure 28). The raw DEM usually have artificial features such as bridges and culverts, blocking the flow on the DEM, though on the ground water can flow through. An easy solution is to manually create a layer of short segments cutting across the bridges or roads. Make sure the segment touches the low areas at both ends.

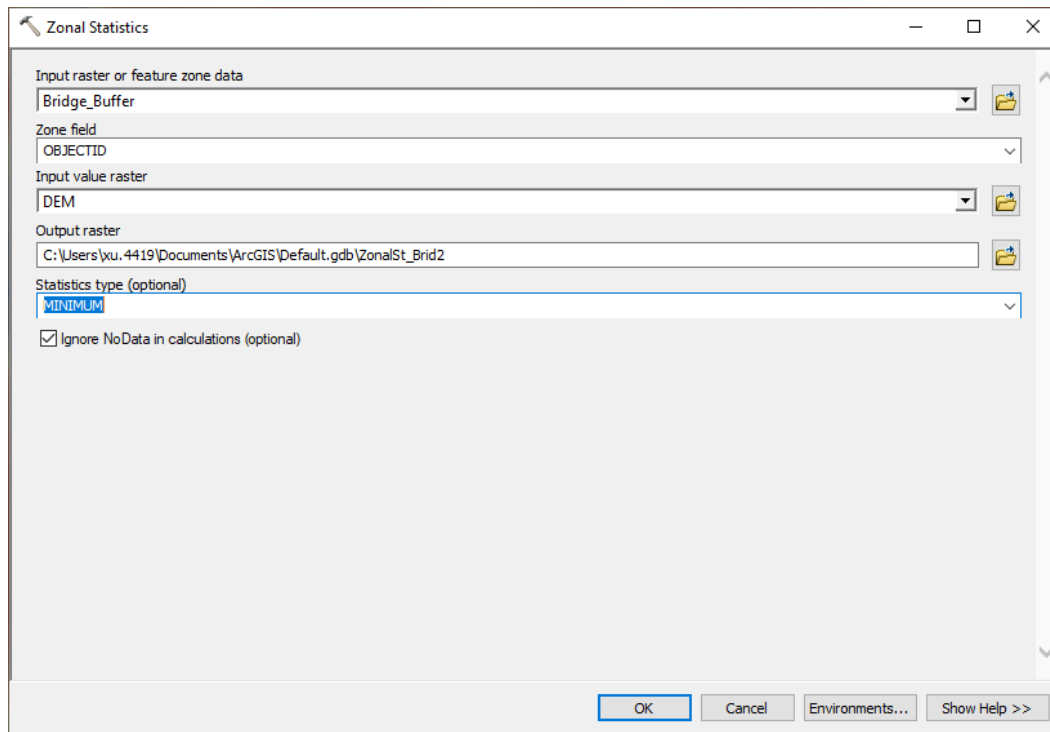


Then, create buffers around the segments (Figure 29). A 2 m buffer distance is large enough. The result is a polygon layer (Figure 30). We will use the polygons to “burn” the DEM.



For each buffer polygon, we need to find the minimum pixel value (lowest elevation), and then replace the original pixels with the minimum pixels. Use the Zonal Statistics tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/zonal-statistics.htm>) and select the buffer layer as the zone data (Figure 31). Then select a unique ID field for the zone field. Make sure the ID field is unique, or errors will occur. The input value raster should be the DEM. Select **MINIMUM** for the statistics type.

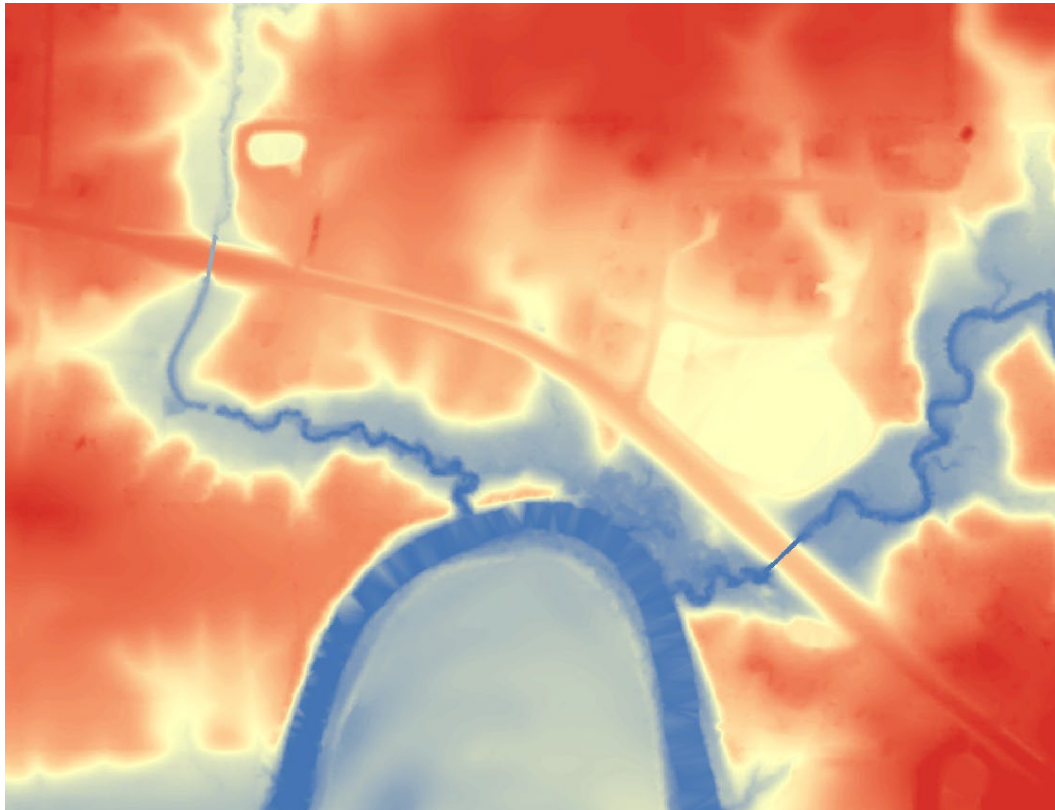
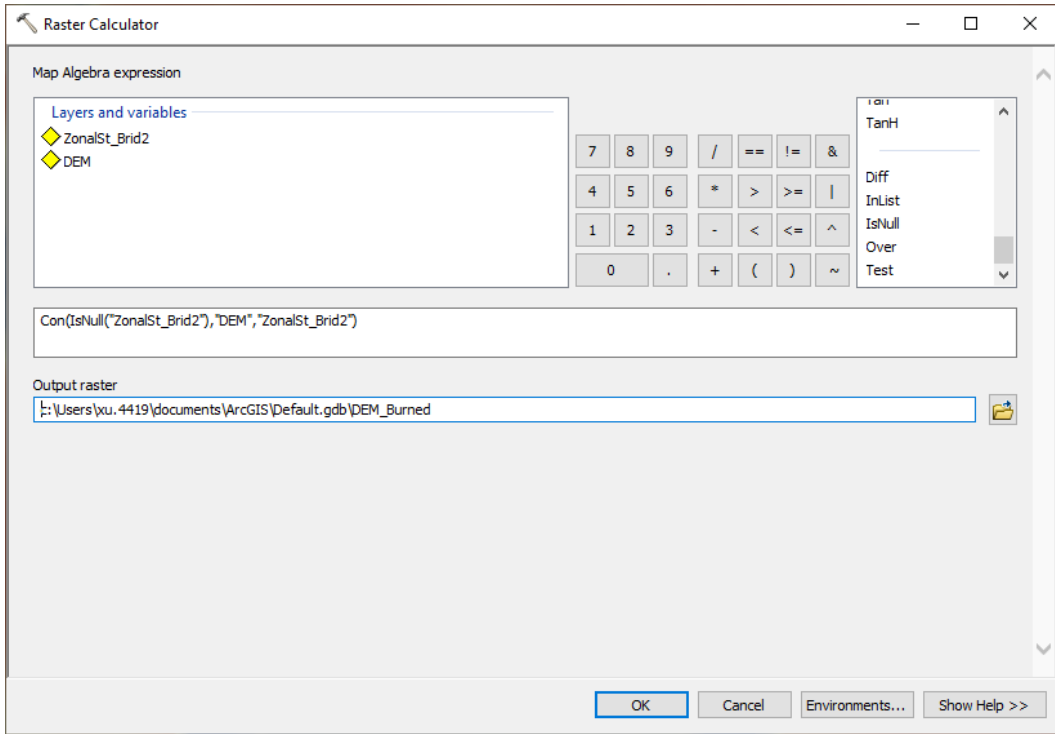
Tip: Remember 1) the *focal statistics* and *zonal statistics* are two different tools, and 2) *zonal statistics* and *zonal statistics as table* are two different tools. If you are unclear, check the ArcGIS help document.



The output of the zonal statistics is a raster layer. The raster is no data anywhere outside the polygons but has the lowest elevation in each polygon, respectively.

Tip: If the ID field is not unique, then multiple (in worst case, all) polygons have the same lowest elevation, which is incorrect.

Next, use Raster Calculator to replace the raw DEM pixels with the zonal minimum values (Figure 32). Remember to set the **Processing Extent** in the **Environmental Settings** to **Union of Inputs**, or the output DEM will miss some boundary areas. The output DEM should have the bridges or culverts removed (Figure 33).

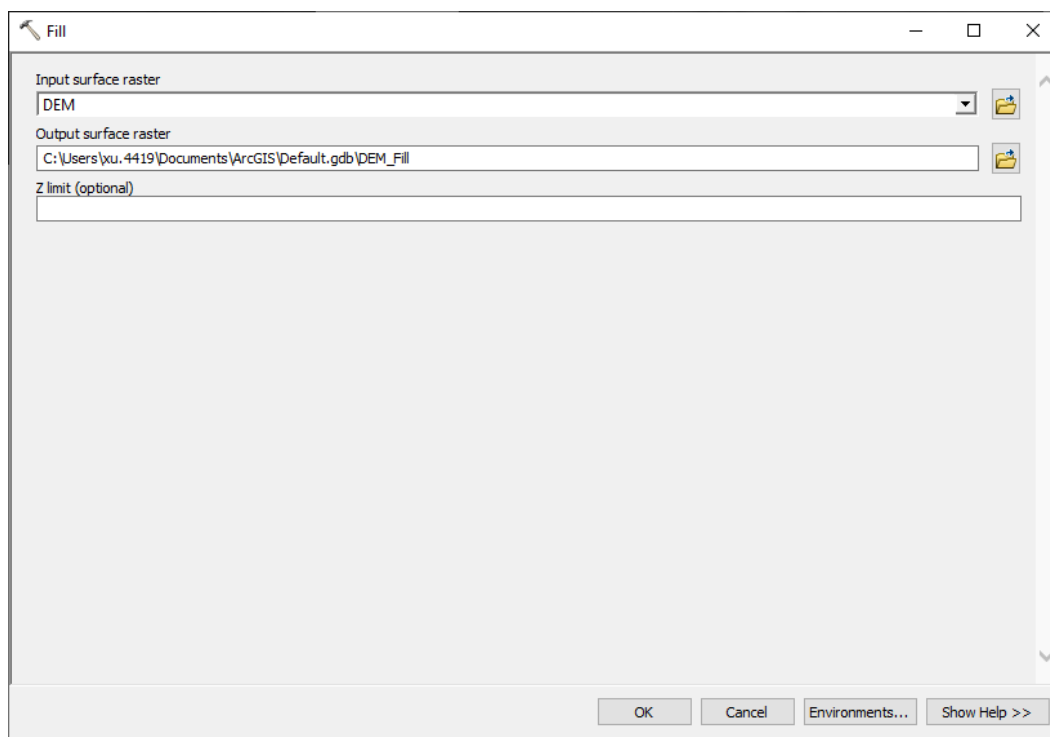


Channel Extraction

The channel extraction uses the standard D8 method. The D8 method is based on topographically driven flow direction and accumulation. D8 requires pre-processing of the DEM, which including breaching the artificial features and filling/breaching the natural depressions (sinks). The above steps of burning the DEM (Figures 28-33) are the breaching of artificial features. Now we will process the natural sinks.

Tip: You must breach the artificial features (bridges, culverts) in the first step, before working on natural sinks. Sinks (digital sinks) created by artificial features should not be filled, but only breached.

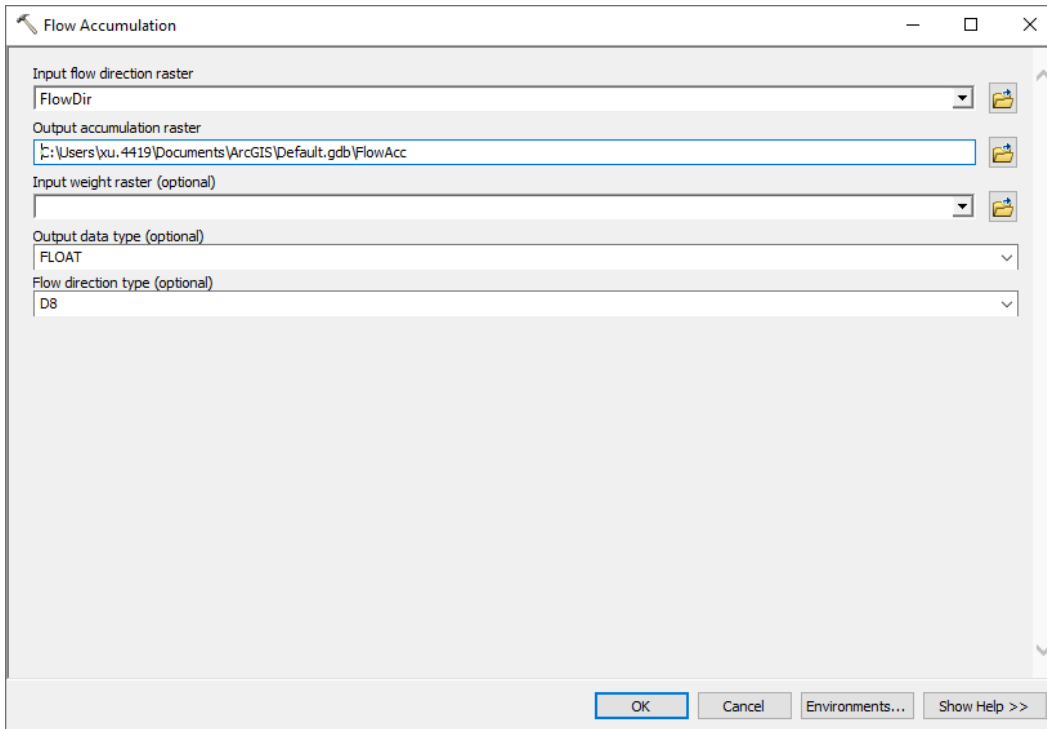
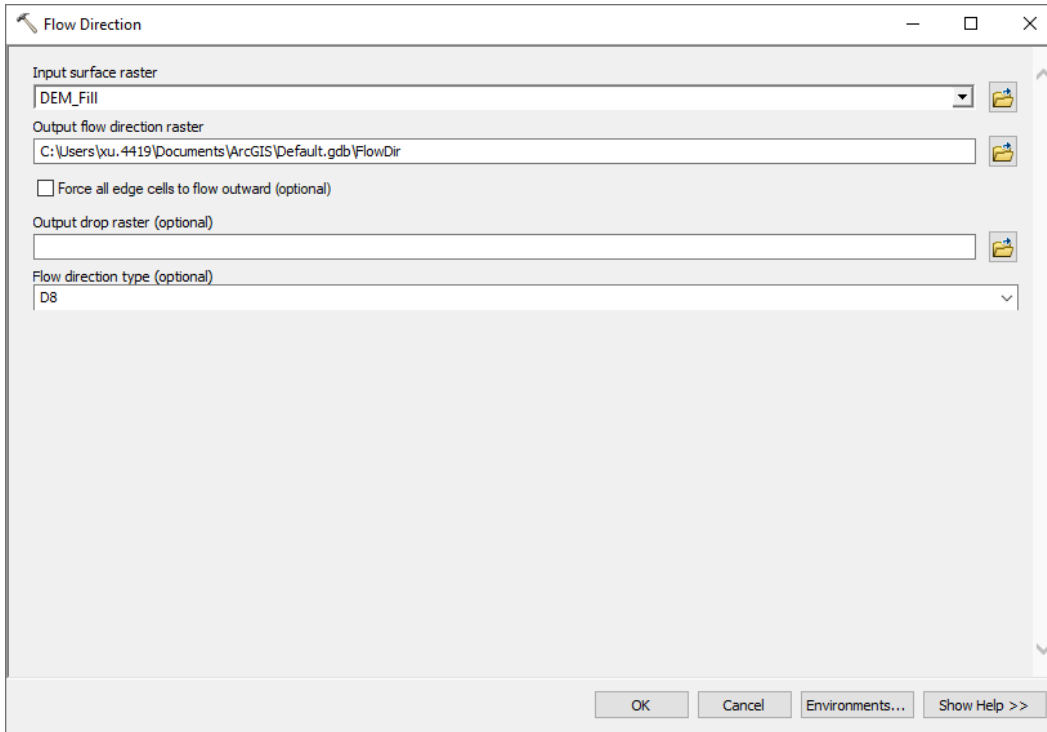
The hydrologically enforced DEM may contain isolated depressions (sinks) where surface runoff cannot correctly flow due to the D8 algorithm, and thus causes errors. Sinks can naturally occur, such as Carolina bays (https://en.wikipedia.org/wiki/Carolina_bays). To fill sinks, use the Fill tool (Figure 34, <https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/fill.htm>).



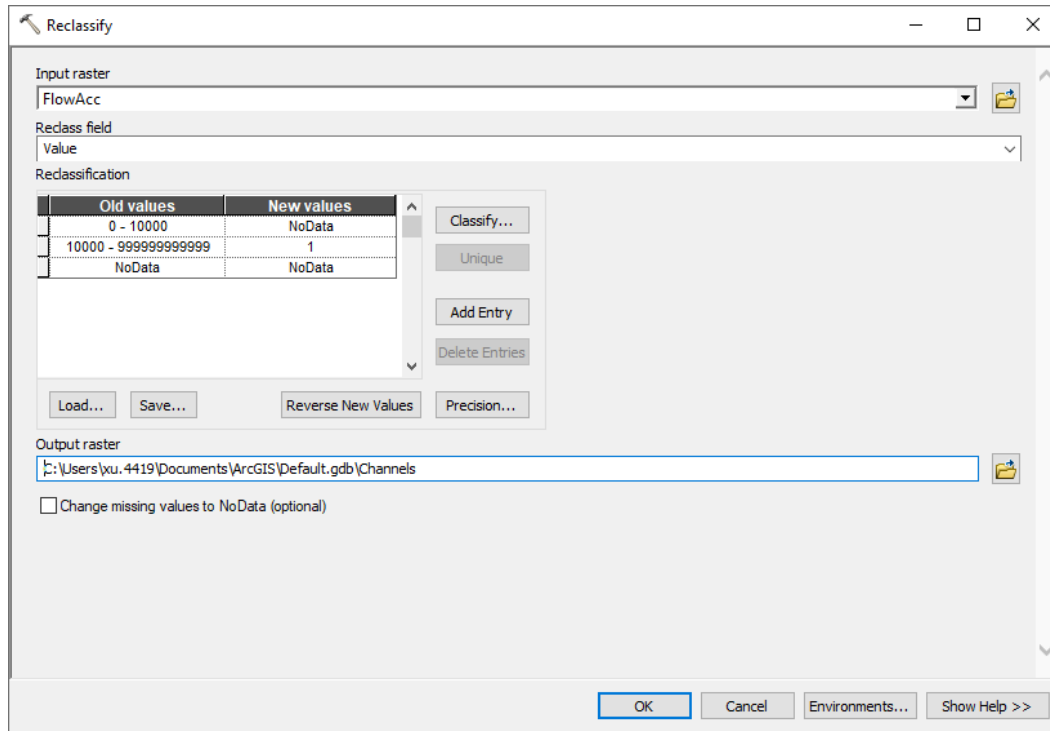
The output of Figure 34 is a filled DEM with correct flow directions.

Next, calculate the flow direction from the filled DEM (Figure 35); the output is a raster layer of flow direction. The tool is at <https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/flow-direction.htm>.

Next, use the flow direction layer as the input and calculate the flow accumulation (Figure 36); the output is raster layer of flow accumulation. Each pixel has a flow accumulation area value, ranging from 0 to a positive number. The Flow Accumulation tool is at <https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/flow-accumulation.htm>.

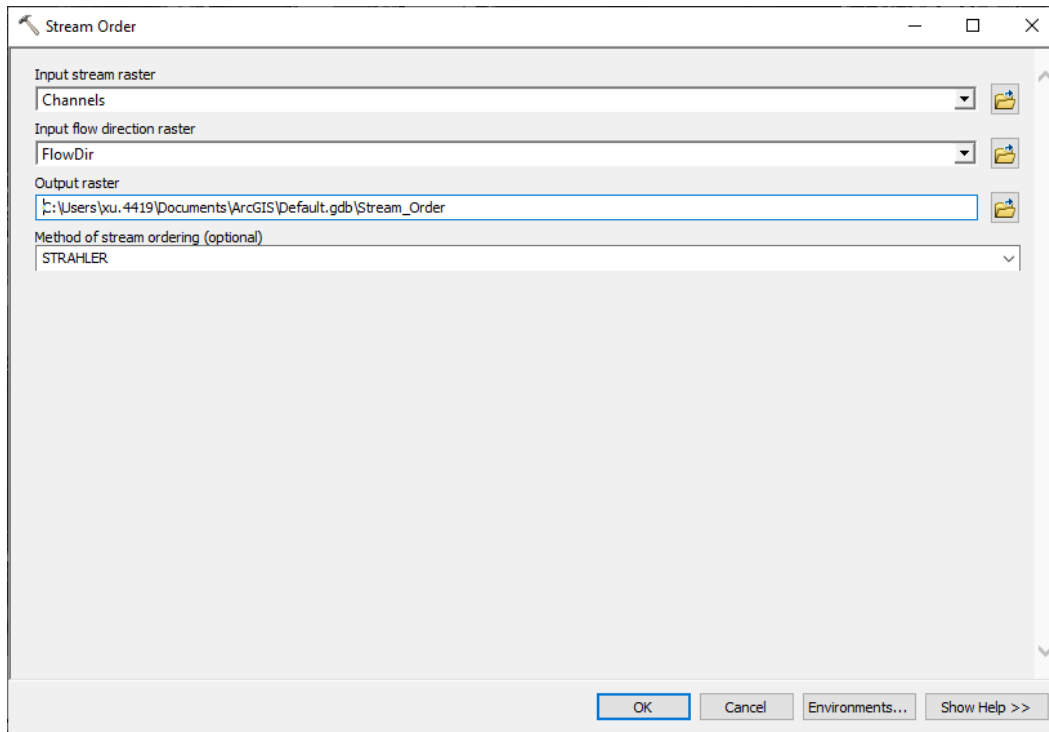


The user will decide a threshold of drainage area for channels. Pixels with flow accumulation below the threshold are considered non-channels; pixels above the threshold represent channels. For simplicity we chose 10000 m² because the result channels largely match those on aerial images (Figure 37). The tool is the Reclassify tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/reclassify.htm>).



The output is a raster layer with only one value (1) representing channels. The channels can be converted to polylines by the Raster to Polyline tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/conversion-toolbox/raster-to-polyline.htm>).

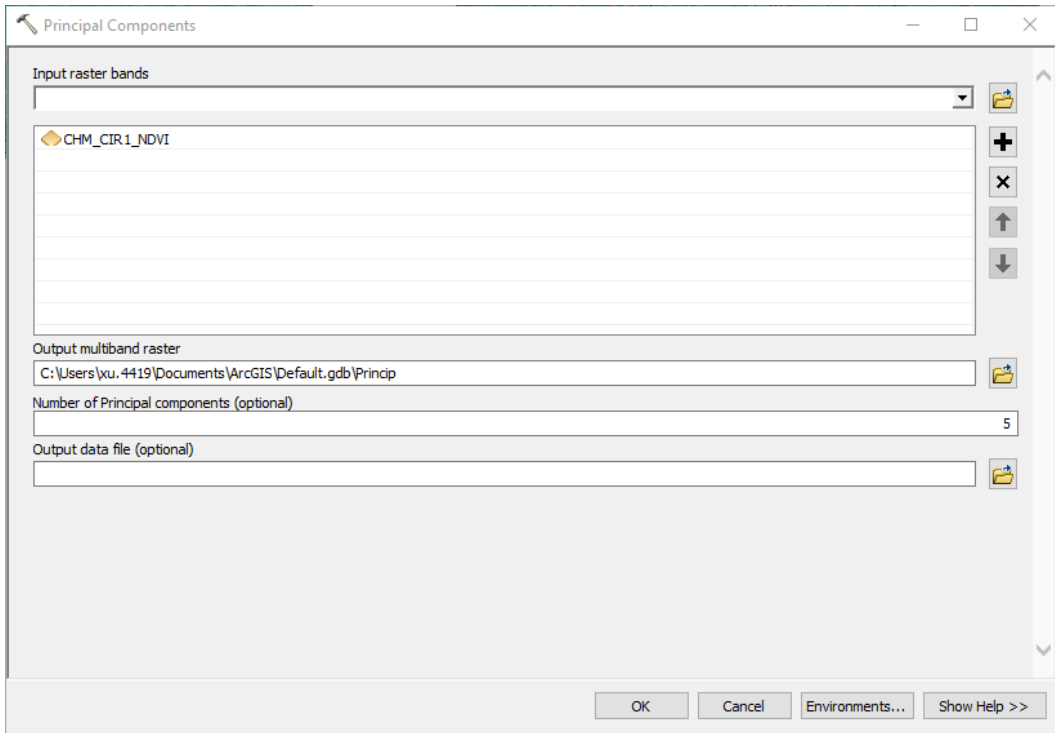
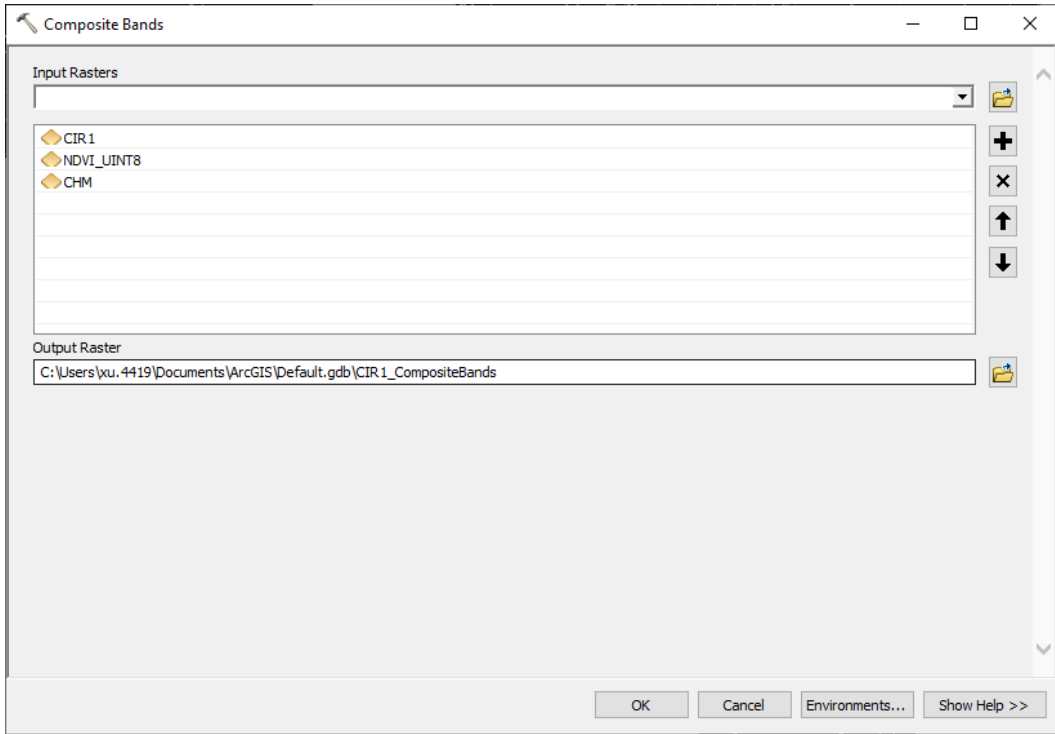
Optionally, you can calculate the stream order such as the Strahler stream order (<https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/stream-order.htm>). The input layers are the flow direction and the reclassified channels (Figure 38).



Creating a Multi-Band Image for Classification

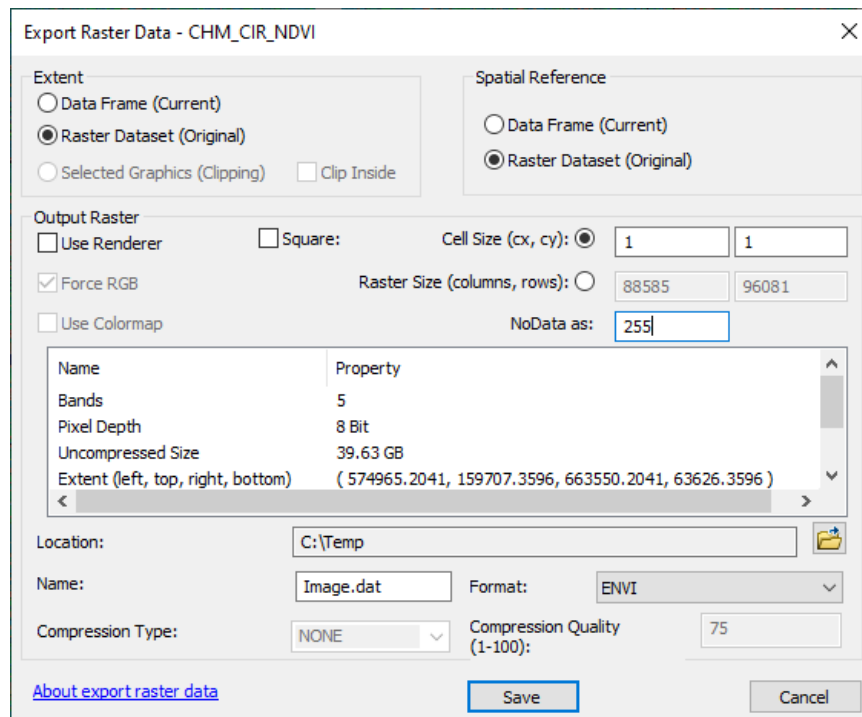
The spectral bands (CIR, true color), NDVI, and CHM are merged to a multi-band image using the Composite Bands tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/data-management-toolbox/composite-bands.htm>, Figure 39). The CHM is in the first place because we will use it as a mask layer; see below for details.

The user may use principal component analysis (PCA) to downsize the image if 1) the number of input band number is too large (e.g., over 20), or 2) the input bands are highly correlated. The tool is found at <https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/principal-components.htm> (Figure 40). We did not use PCA because the band number rarely exceeds ten, and the information is usually independent from each other (e.g., optical image and CHM).



Finally, export the multi-band image to a local folder in ENVI format (Figure 41). The ENVI image file format is uncompressed and hence, easy for custom programming. The ENVI header file is also easy to read and edit (<https://www.l3harrisgeospatial.com/docs/enviheaderfiles.html>). Use 255 for the NoData value.

Tip: The reason to use 255 for NoData is to reduce the computation time. NoData areas around a study site may be as much as the study site area with data. We are not interested in those boundary pixels and will use a mask layer to identify them. An easy solution without creating more data is to use the CHM layer. A boundary pixel will have a 255 CHM value, and it will not be processed. A valid pixel in the study site has a lower CHM value, so it will be processed.



Training

The biggest challenge in selecting training samples comes from forested and emergent wetlands. Here we use the ancillary information from NWI (<https://www.fws.gov/program/national-wetlands-inventory>) and use the existing wetland polygons as training samples. But we do not want to arbitrarily select NWI polygons, instead, we will randomize this process.

The NWI for the entire state contains too many features. First, clip the NWI by the study site boundary, using the Clip tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/analysis-toolbox/clip.htm>). This Clip tool is for vector layers, different than the Clip Raster tool.

The NWI usually contains all wetland types, but we are only interested in the forested and emergent wetlands. Thus, remove all other wetland types. If the shapefile (feature class) has millions of features,

then selecting the target wetland features and exporting them as a separate layer is usually faster than editing the original layer.

Next, add a new field in the attribute table (Figure 42). This field is to be assigned a random number. A possible solution is to export the attribute table, open in Excel, add the random numbers (<https://support.microsoft.com/en-us/office/rand-function-4cbfa695-8869-4788-8d90-021ea9f5be73>), and join the table back to the NWI shapefile. You will need a unique ID field for this joining. Use the ArcGIS join tool (<https://desktop.arcgis.com/en/arcmap/latest/manage-data/tables/joining-attributes-in-one-table-to-another.htm>).

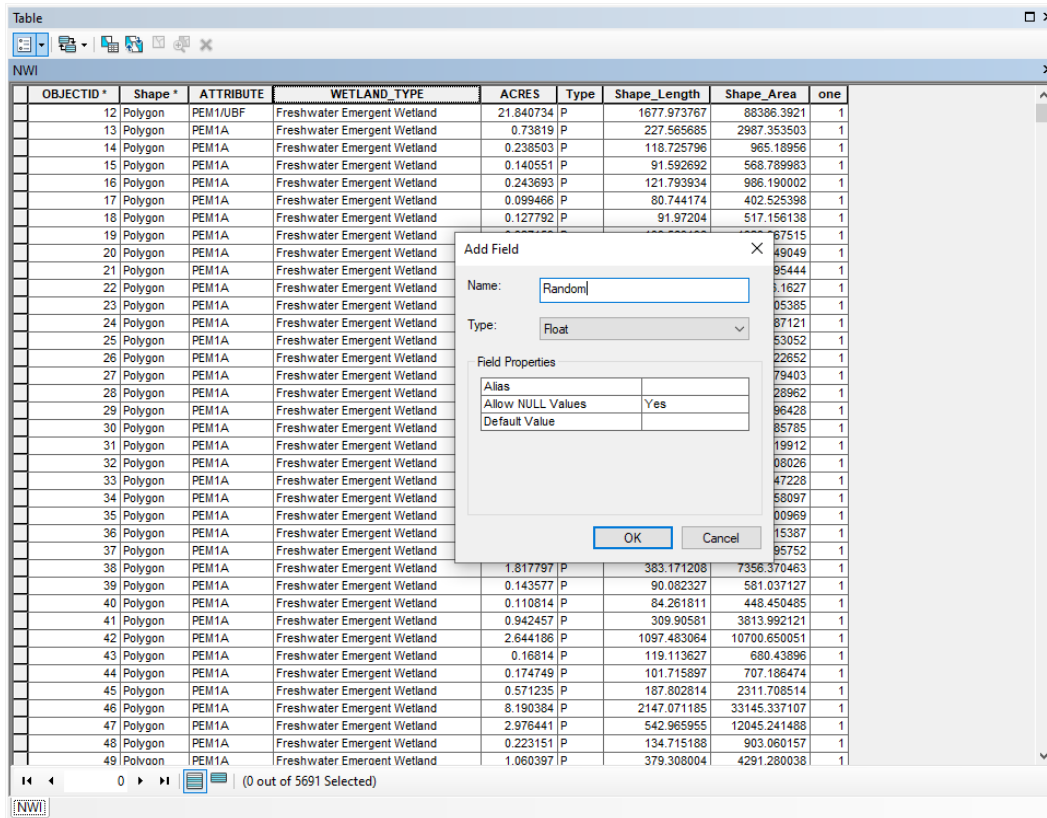


Figure 42. Adding a random number field to the NWI layer.

Now update the Random field with the random number generated in Excel. Then, remove the join.

Rank the random number field and select forested and emergent wetland samples from the smallest or the biggest random number, until the enough samples are selected. The NWI may contain errors, so be critical when selecting training samples. If an NWI sample is not obvious on images, it can be skipped. You don't need to use the exact shape of an NWI polygon, because they can often be too large. Later we will convert the polygons to pixels, and a 32 x 32 m polygon contain over 1000 pixels. It is also recommended that all training samples are of similar sizes, so that all classes have equal weight.

Other classes are easy to identify on images. Figures 43-45 show some training samples. Use a short int field for the class labels, starting from 1. For example, all open water samples should have the same class label (here is 3 in Figure 44).

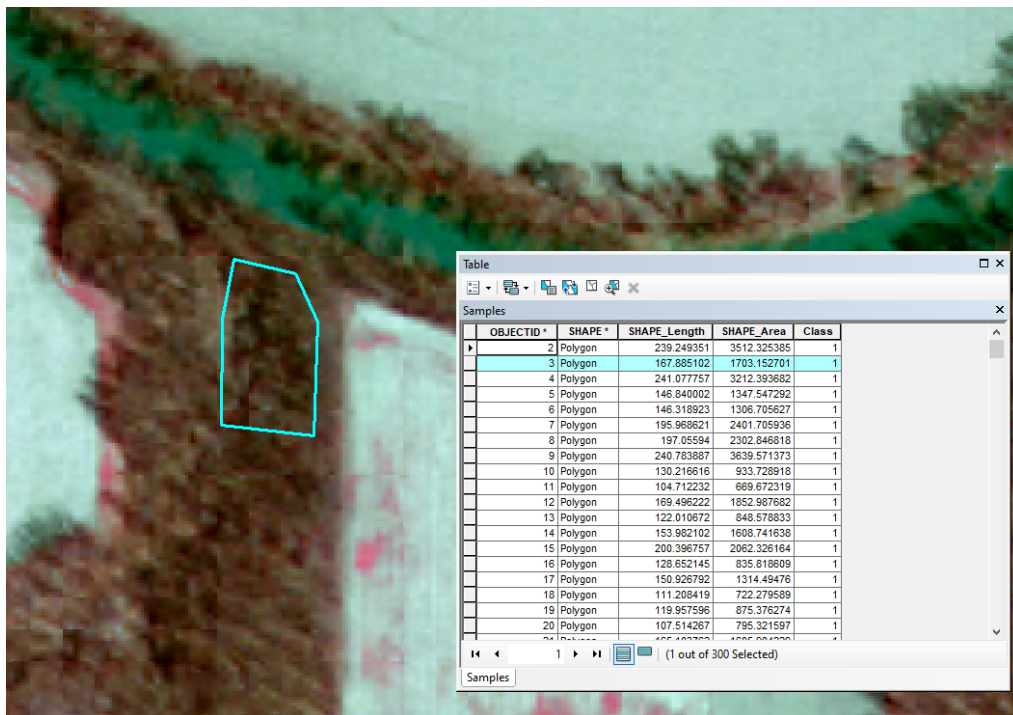


Figure 43. A sample of forested wetland.

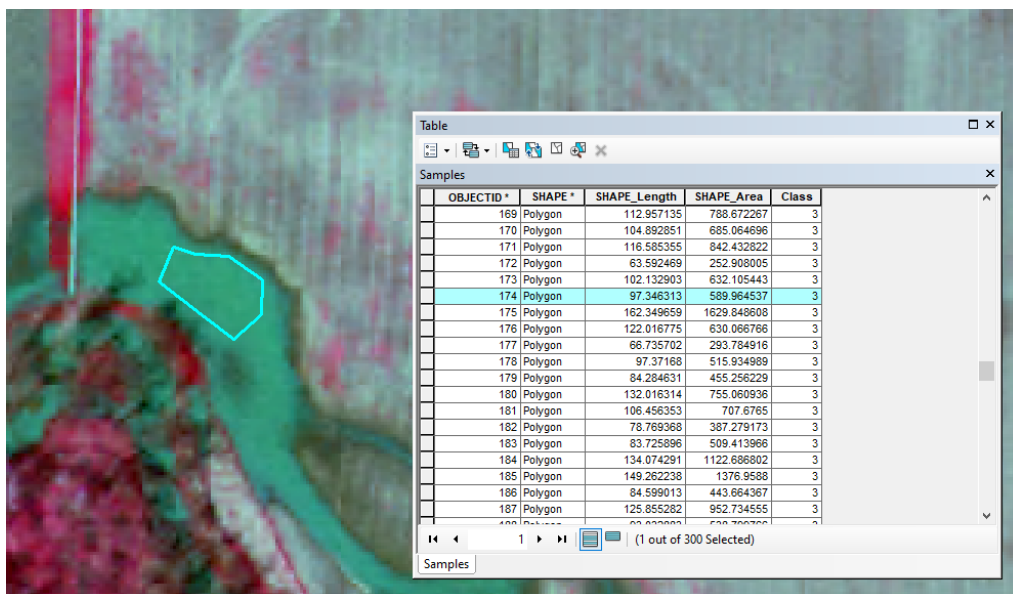


Figure 44. A sample of open water.

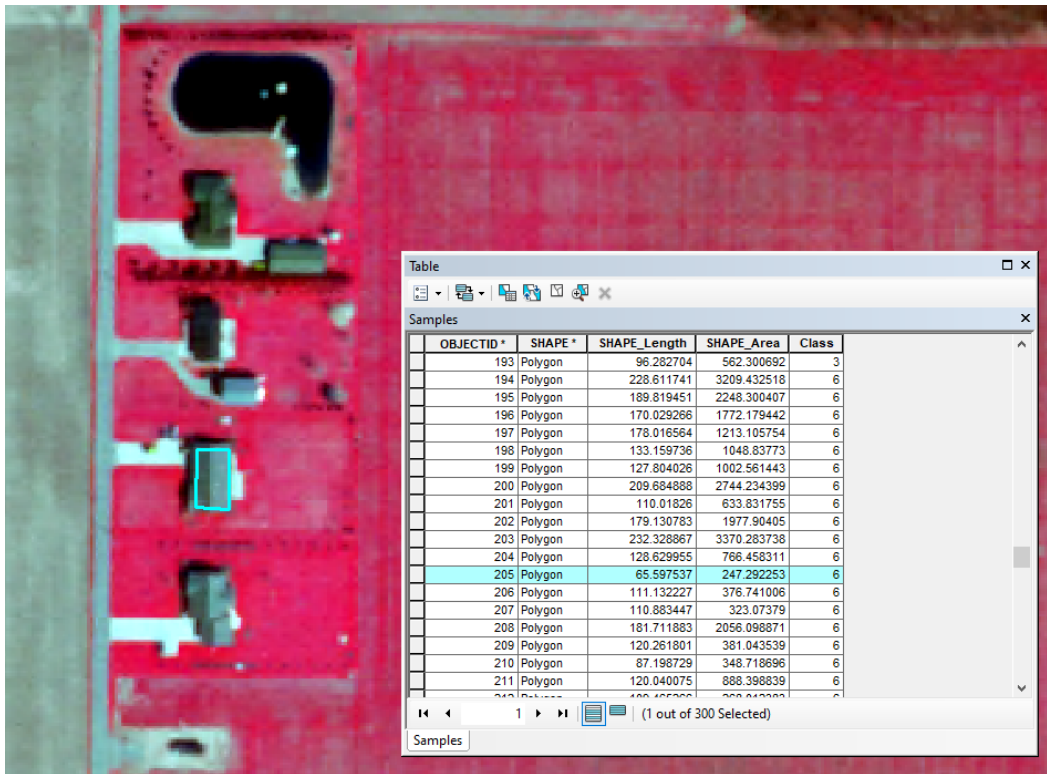


Figure 45. A sample of developed land.

We will use the pixel information inside the polygons. Convert the sample polygons to a raster layer using Polygon to Raster tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/conversion-toolbox/polygon-to-raster.htm>). The value field should be the class label (Figure 46). For the cell size, you can use the raster resolution (1 m) or a coarser resolution (e.g., 2 m in Figure 46). Using a coarser resolution will downsize the training set and may be preferable if the training polygons are large.

The output of Figure 46 is a single raster layer with valid pixel values (class labels) inside the training polygons. Outside the polygons, the raster is NoData. Next, convert the raster layer to points using the Raster to Point tool (Figure 47, <https://desktop.arcgis.com/en/arcmap/latest/tools/conversion-toolbox/raster-to-point.htm>). The class labels will be included in the grid_code field (Figure 48).

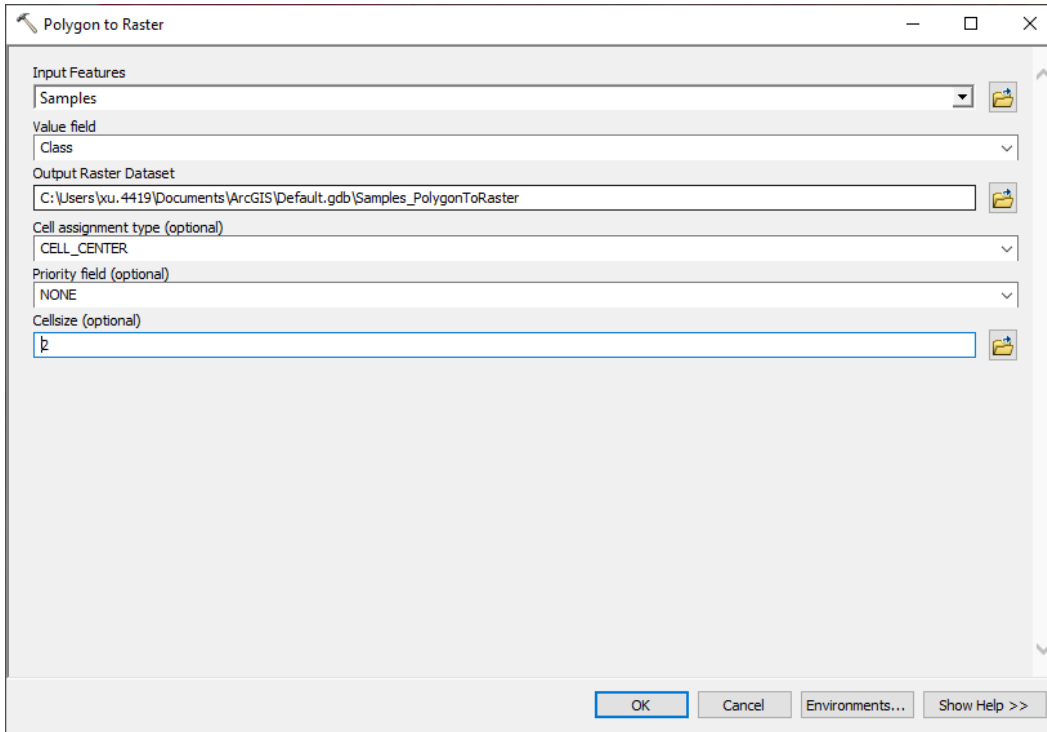


Figure 46. Converting sample polygons to raster.

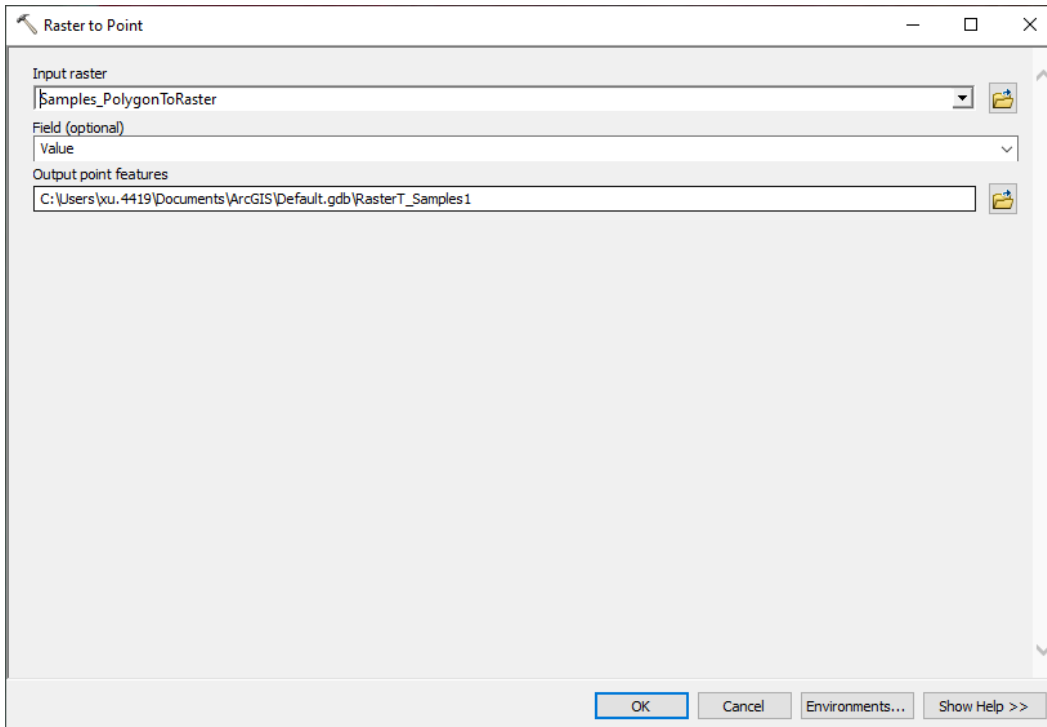


Figure 47. Converting the raster to points.

OBJECTID *	Shape *	pointid	grid_code
1	Point	1	1
2	Point	2	1
3	Point	3	1
4	Point	4	1
5	Point	5	1
6	Point	6	1
7	Point	7	1
8	Point	8	1
9	Point	9	1
10	Point	10	1
11	Point	11	1
12	Point	12	1
13	Point	13	1
14	Point	14	1
15	Point	15	1
16	Point	16	1
17	Point	17	1
18	Point	18	1
19	Point	19	1
20	Point	20	1
21	Point	21	1
22	Point	22	1
23	Point	23	1
24	Point	24	1
25	Point	25	1
26	Point	26	1
27	Point	27	1
28	Point	28	1

Figure 48. The grid_code field contains the class labels.

Now we have a point (vector) layer of training samples. The only attribute information is the class label. To add spectral information from the multi-band image, use the Extract Multi Values to Points tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/spatial-analyst-toolbox/extract-multi-values-to-points.htm>) and select the multi-band image prepared in the previous steps (Figure 49).

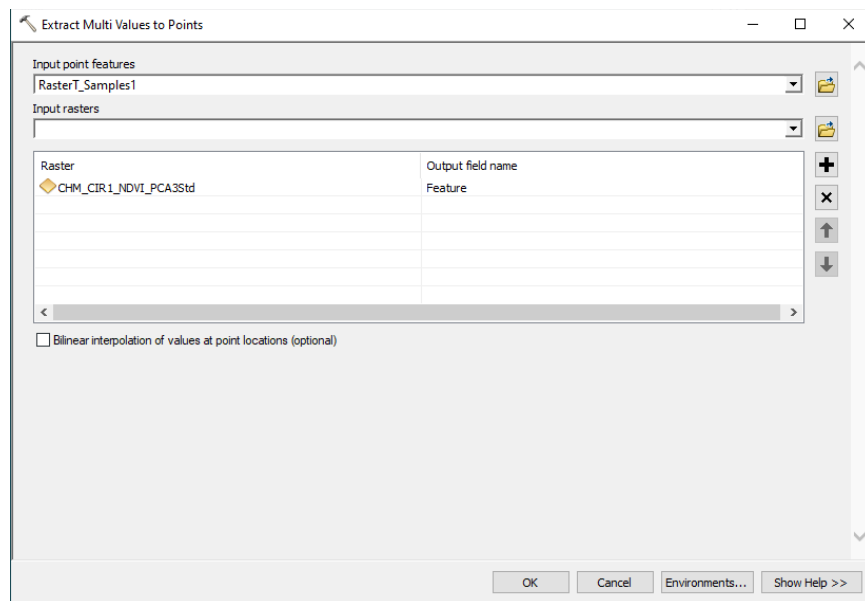


Figure 49. Extracting spectral information to the points.

OBJECTID *	Shape *	pointid	grid_code	b1_Feature	b2_Feature	b3_Feature	b4_Feature	b5_Fea ^
1	Point	161917	5	0	92	80	72	
2	Point	161918	5	0	87	77	70	
3	Point	161919	5	0	89	77	70	
4	Point	161920	5	0	87	76	69	
5	Point	161921	5	0	84	76	68	
6	Point	161922	5	0	85	76	69	
7	Point	161923	5	0	90	79	71	
8	Point	161924	5	0	90	80	73	
9	Point	161925	5	0	91	79	72	
10	Point	161926	5	0	89	78	71	
11	Point	161927	5	0	84	76	69	
12	Point	161928	5	0	84	78	70	
13	Point	161929	5	0	84	77	69	
14	Point	161930	5	0	86	77	70	
15	Point	161931	5	0	79	68	62	
16	Point	161932	5	0	82	72	64	
17	Point	161933	5	0	84	76	67	
18	Point	161934	5	0	84	76	68	
19	Point	161935	5	0	82	76	68	
20	Point	161936	5	0	81	76	69	
21	Point	161937	5	0	84	78	71	
22	Point	161938	5	0	87	78	70	
23	Point	161939	5	0	91	82	74	
24	Point	161940	5	0	92	82	74	
25	Point	161941	5	0	89	82	75	
26	Point	161942	5	0	88	82	76	
27	Point	161943	5	0	88	80	73	

Figure 50. The points now have the spectral information.

The points now will have the spectral information (Figure 50). The first band (b1_Feature in Figure 50) is the CHM band and the class label 5 is non-forested upland. Hence, the value is typically zero. The next three bands are the spectral bands of the aerial image.

Export the attribute table in Figure 50 to a local file (e.g., text file). Copy the content to Excel and organize the columns to match the format in Figure 51. Here the Y field is the class label and A through H are the attribute fields.

The screenshot shows an Excel spreadsheet with the following data:

	A	B	C	D	E	F	G	H	I	J	Y
1	0	92	80	72	136	4	5	1	5		
2	0	87	77	70	135	2	2	1	5		
3	0	89	77	70	136	2	2	1	5		
4	0	87	76	69	136	2	3	1	5		
5	0	84	76	68	133	1	2	1	5		
6	0	85	76	69	134	2	2	1	5		
7	0	90	79	71	135	4	5	1	5		
8	0	90	80	73	135	2	2	1	5		
9	0	91	79	72	136	3	3	1	5		
10	0	89	78	71	135	3	4	1	5		
11	0	84	76	69	133	4	3	1	5		
12	0	84	78	70	132	3	2	1	5		
13	0	84	77	69	133	3	2	1	5		
14	0	86	77	70	134	5	3	0	5		
15	0	79	68	62	137	4	3	0	5		
16	0	82	72	64	135	3	2	1	5		
17	0	84	76	67	133	2	2	1	5		
18	0	84	76	68	133	1	2	1	5		
19	0	82	76	68	132	2	2	1	5		
20	0	81	76	69	131	3	3	1	5		
21	0	84	78	71	132	5	5	1	5		
22	0	87	78	70	134	4	3	1	5		
23	0	91	82	74	134	2	3	1	5		
24	0	92	82	74	134	3	3	1	5		
25	0	89	82	75	132	3	3	1	5		
26	0	88	82	76	132	3	2	1	5		
27	0	88	82	76	132	3	2	1	5		
28	0	88	82	76	132	3	2	1	5		

Figure 51. Formatting the training samples in Excel.

Next, export the training samples from Excel to a local text file and save it (Figure 52).

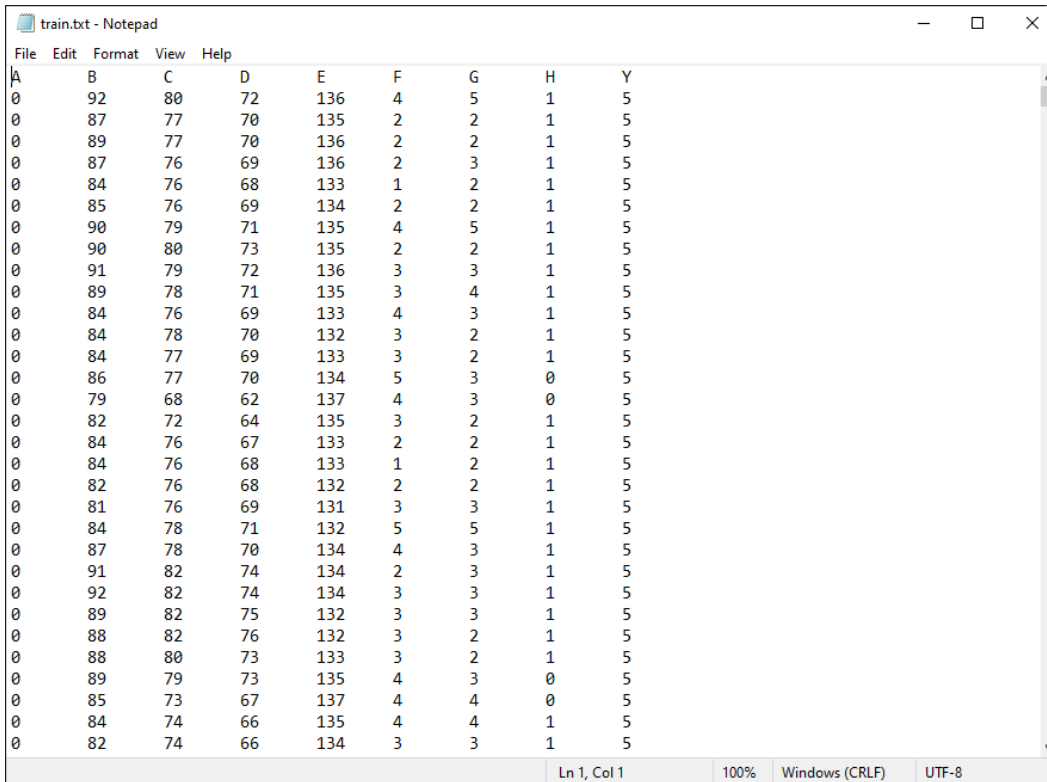


Figure 52. The training samples saved as a local text file.

Now open MATLAB and load the training samples (Figure 53). To load data, click the “Import Data” button under the “HOME” tab, and select the text file. Then a dialog will pop up as shown in Figure 53. Keep everything as default and click the green check mark above “Import Selection”.

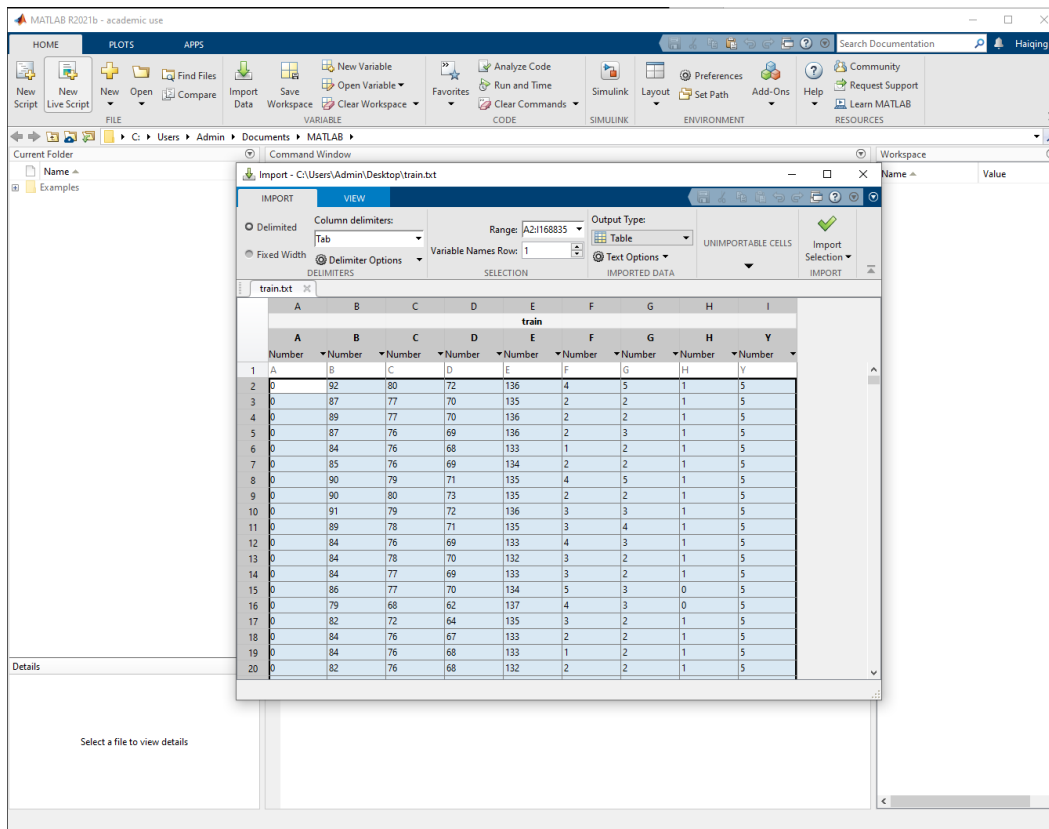


Figure 53. Importing the training samples to MATLAB.

Now you can verify that the training samples have been loaded to MATLAB (Figure 54). The data is stored as a table with the variable name “train”.

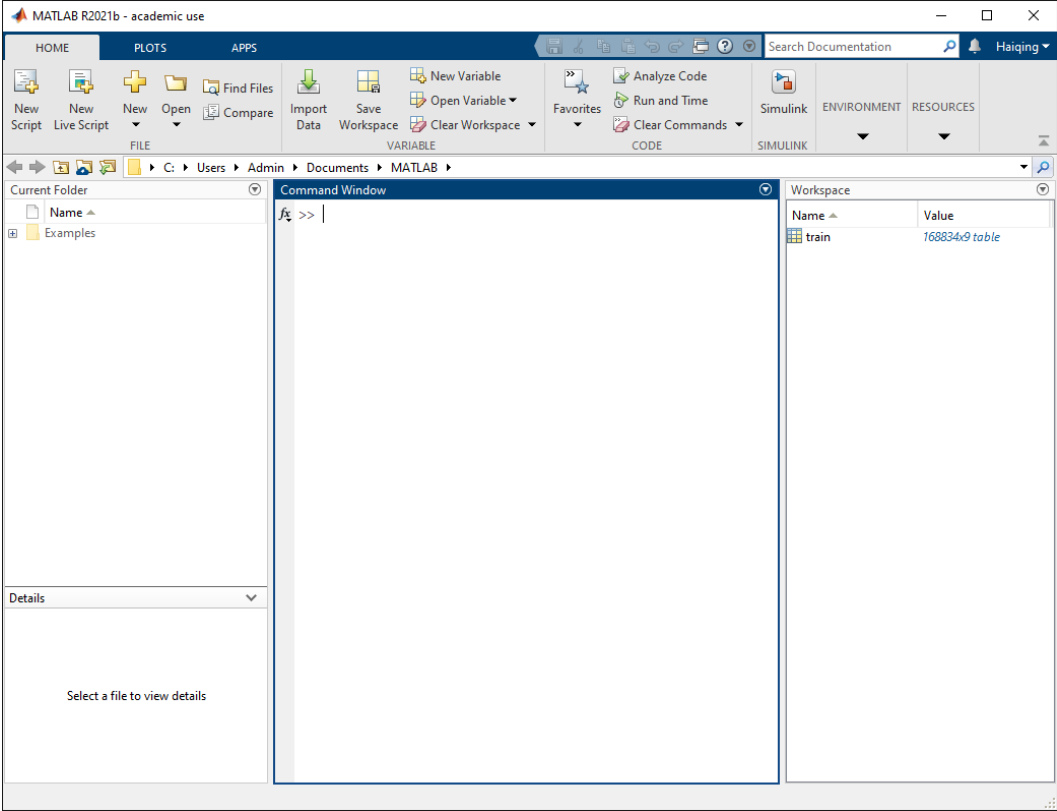


Figure 54. Training samples have been loaded to MATLAB.

Now we will train the samples. Click the “APPS” tab and find the “Classification Learner”. A new dialog will pop up. On the new dialog, expand “New Session” and then click “From Workspace” (Figure 55). Keep every parameter as default and start the session.

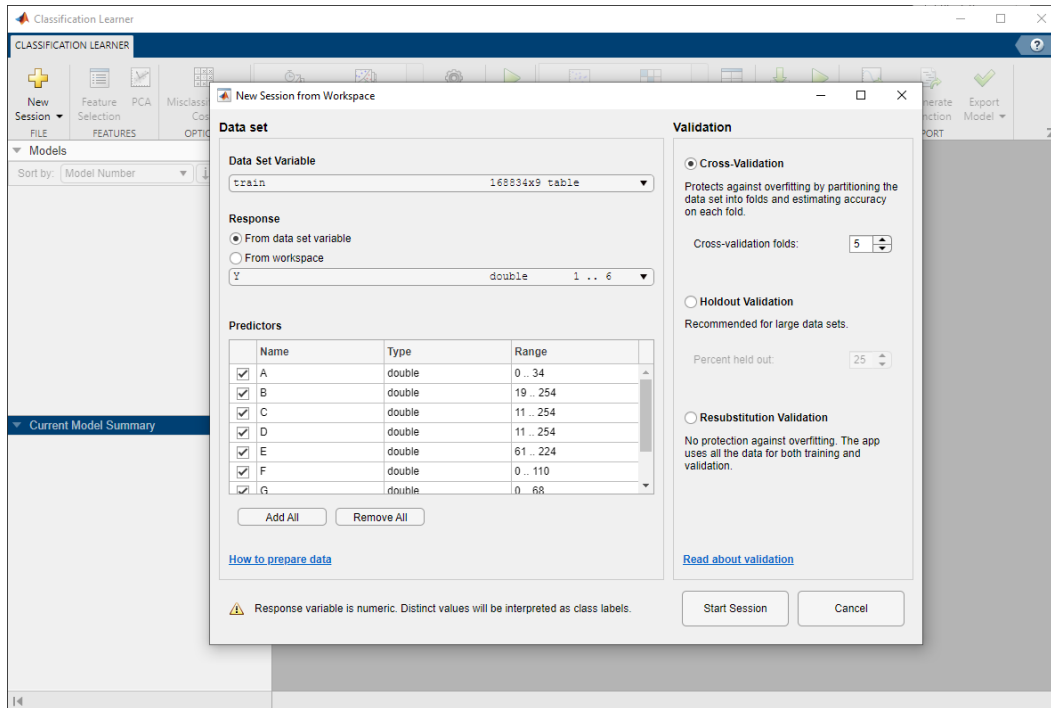


Figure 55. Importing the training samples to Classification Learner.

We will use Bagged Trees as the classifier and train the samples (Figure 56). Locate “MODEL TYPE” and expand the list, and then choose “Bagged Trees” (Figure 56). Next, click the “Train” button with the green triangle. Training may take a few minutes.

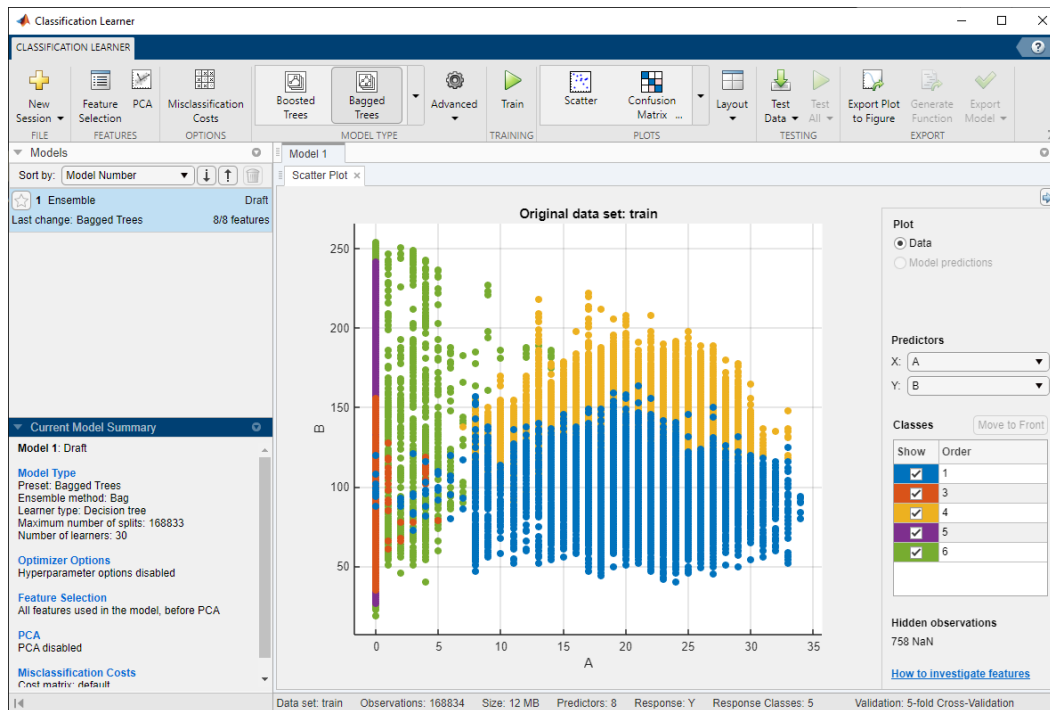


Figure 56. Training the samples in MATLAB.

When the training is complete, you will see a confusion matrix (error matrix) at the center of the dialog (Figure 57). The left panel shows an accuracy, e.g., 96.2% in Figure 57. This accuracy is the training accuracy. Ideally, this accuracy should be high enough, so that different classes are easier to separate. However, a high training accuracy does not guarantee a high accuracy validated with an independent testing samples.

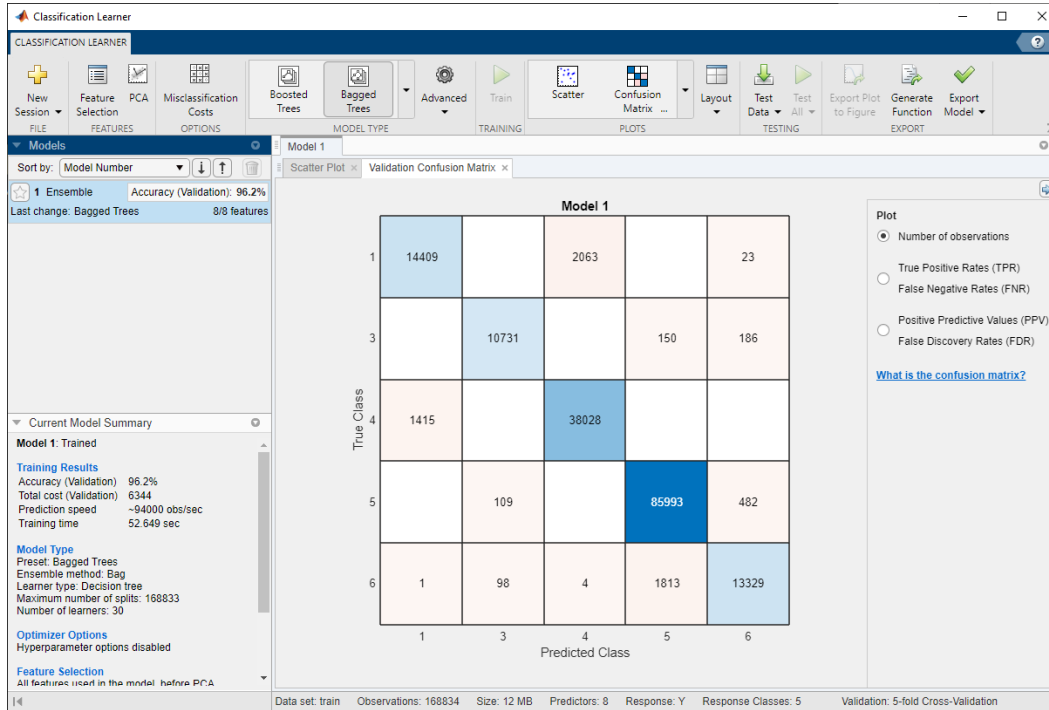


Figure 57. The trained classification model.

To export the trained model, expand the “Export Model” option, and choose “Export Compact Model”. Name the output model “BaggedTrees”. After this step, you can close the Classification Learner dialog (but not the MATLAB software).

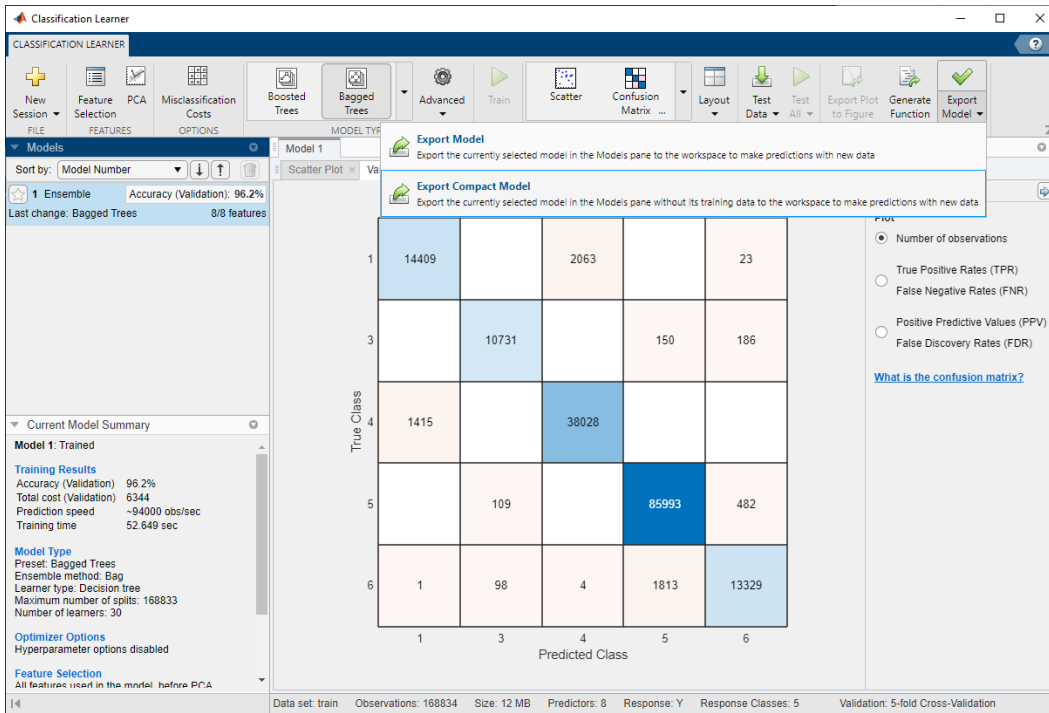


Figure 58. Exporting the trained model.

Now the trained model “BaggedTrees” is still stored in the computer RAM (Figure 59). Right click this variable and click “Save as” and save it to a local file with the name “BaggedTrees.dat”. After this step, you can close the MATLAB software.

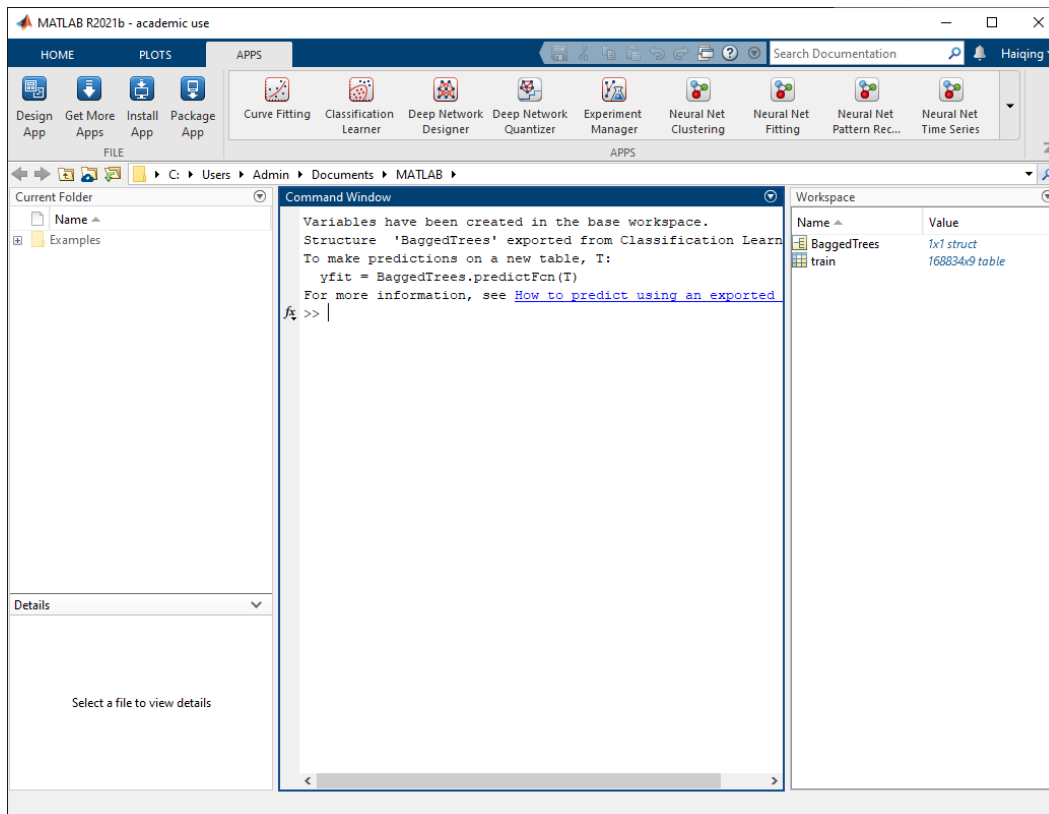
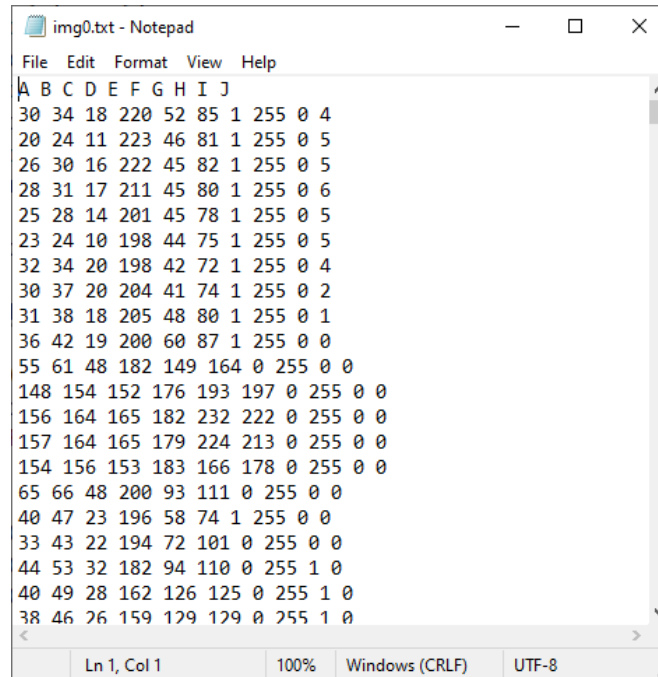


Figure 59. Exporting the trained model to a local file.

Classification

To classify the entire study site, we need to convert the multi-band image to text format. The desired format should have all spectral bands except the class label (Figure 60). We will do this automatically with a custom C++ program (Figure 61). The first button reads the multi-band ENVI image (the output of Figure 41) and converts it to multiple text files. The image is partitioned to multiple smaller text files determined by the block size (Figure 61).



```
File Edit Format View Help
A B C D E F G H I J
30 34 18 220 52 85 1 255 0 4
20 24 11 223 46 81 1 255 0 5
26 30 16 222 45 82 1 255 0 5
28 31 17 211 45 80 1 255 0 6
25 28 14 201 45 78 1 255 0 5
23 24 10 198 44 75 1 255 0 5
32 34 20 198 42 72 1 255 0 4
30 37 20 204 41 74 1 255 0 2
31 38 18 205 48 80 1 255 0 1
36 42 19 200 60 87 1 255 0 0
55 61 48 182 149 164 0 255 0 0
148 154 152 176 193 197 0 255 0 0
156 164 165 182 232 222 0 255 0 0
157 164 165 179 224 213 0 255 0 0
154 156 153 183 166 178 0 255 0 0
65 66 48 200 93 111 0 255 0 0
40 47 23 196 58 74 1 255 0 0
33 43 22 194 72 101 0 255 0 0
44 53 32 182 94 110 0 255 1 0
40 49 28 162 126 125 0 255 1 0
38 46 26 159 129 129 0 255 1 0
```

Figure 60. The multi-band image has been converted to the text file format.

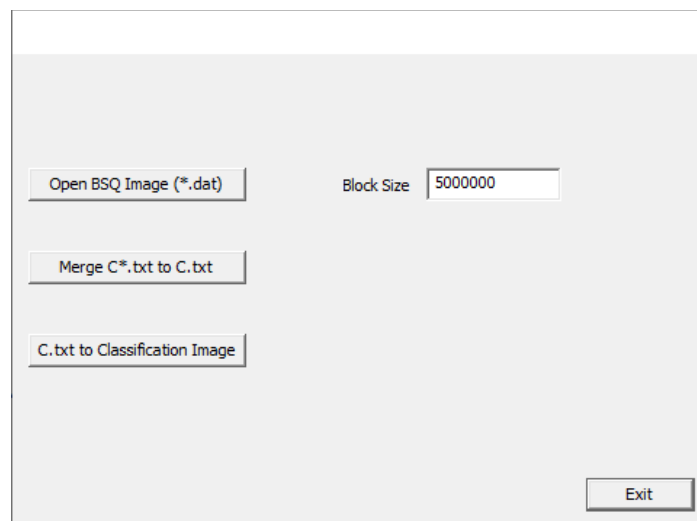
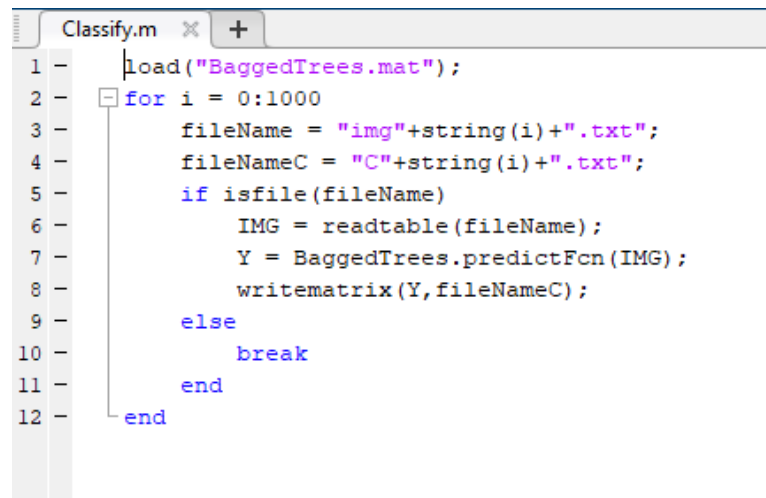


Figure 61. A custom C++ program for converting image and text files.

Now the ENVI image has been converted to multiple (e.g., 100) smaller text files. These files are named as "img*.txt". Here "*" represents the number of the block and it starts from 0. We will classify every image file using the trained model (output of Figure 59). Put all the image text files, and trained model "BaggedTrees.dat", and the MATLAB script file "Classify.m" in the same folder.

Click the script file and you should see the codes (Figure 62). Run the script, and it will classify the image blocks. The output files are named as "C*.txt". Here "*" represents the number of the block and it starts from 0.



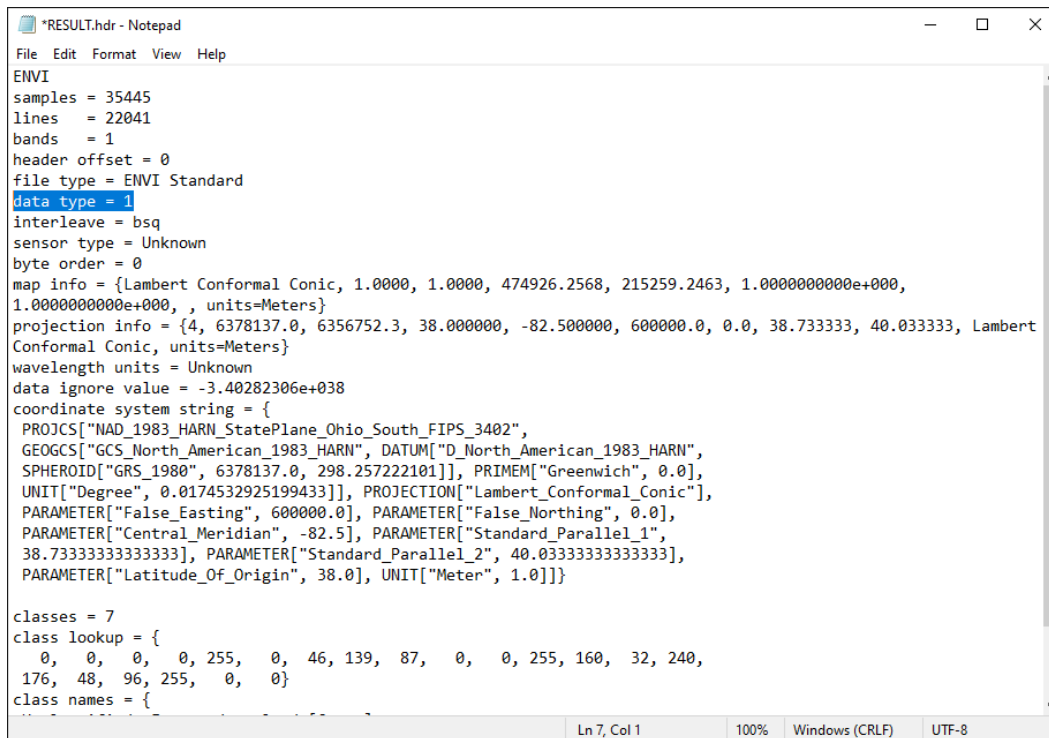
```
1 - load("BaggedTrees.mat");
2 - for i = 0:1000
3 -     fileName = "img"+string(i)+".txt";
4 -     fileNameC = "C"+string(i)+".txt";
5 -     if isfile(fileName)
6 -         IMG = readtable(fileName);
7 -         Y = BaggedTrees.predictFcn(IMG);
8 -         writematrix(Y, fileNameC);
9 -     else
10 -         break
11 -     end
12 - end
```

Figure 62. A custom MATLAB script for image classification.

Once the classification is finished (this may take 24 hours or longer), close the MATLAB software. We will merge the individual classification outputs to a single file. Make sure the C++ program is in the same folder, and run the C++ program, and click the second button "Merge C*.txt to C.txt". When this finishes, you will find a new file "C.txt", the combination of all C*.txt files.

The final step is to convert C.txt back to an image file. Click the third button of the C++ program “C.txt to Classification Image”. The program will ask for an input image; choose the ENVI image as the output of Figure 41. The program will also read C.txt and convert it to binary format.

The final output is a file named “RESULT.dat”. This image file lacks a header file, but we know it has the exact same dimension and map projection as the ENVI image (output of Figure 41). Thus, locate the header file of the ENVI image (.hdr), and make a copy of it. Rename the copied header file to “RESULT.hdr”. Then, open this header file with a text editor (Figure 63). Make sure the data type has the value of 1. Check the ENVI help document for more information about the header file (<https://www.l3harrisgeospatial.com/docs/enviheaderfiles.html>).



```
*RESULT.hdr - Notepad
File Edit Format View Help
ENVI
samples = 35445
lines = 22041
bands = 1
header offset = 0
file type = ENVI Standard
data type = 1
interleave = bsq
sensor type = Unknown
byte order = 0
map info = {Lambert Conformal Conic, 1.0000, 1.0000, 474926.2568, 215259.2463, 1.0000000000e+000,
1.0000000000e+000, , units=Meters}
projection info = {4, 6378137.0, 6356752.3, 38.000000, -82.500000, 600000.0, 0.0, 38.733333, 40.033333, Lambert
Conformal Conic, units=Meters}
wavelength units = Unknown
data ignore value = -3.40282306e+038
coordinate system string = {
PROJCS["NAD_1983_HARN_StatePlane_Ohio_South_FIPS_3402",
GEOGCS["GCS_North_American_1983_HARN", DATUM["D_North_American_1983_HARN",
SPHEROID["GRS_1980", 6378137.0, 298.257222101]], PRIMEM["Greenwich", 0.0],
UNIT["Degree", 0.0174532925199433]], PROJECTION["Lambert_Conformal_Conic"],
PARAMETER["False_Easting", 600000.0], PARAMETER["False_Northing", 0.0],
PARAMETER["Central_Meridian", -82.5], PARAMETER["Standard_Parallel_1",
38.73333333333333], PARAMETER["Standard_Parallel_2", 40.03333333333333],
PARAMETER["Latitude_Of_Origin", 38.0], UNIT["Meter", 1.0]]}

classes = 7
class lookup = {
0, 0, 0, 0, 255, 0, 46, 139, 87, 0, 0, 255, 160, 32, 240,
176, 48, 96, 255, 0, 0}
class names = {
```

Figure 63. Modify the ENVI header file for the classification map.

Predicting Wetland Impacts by Transportation Projects

It is straightforward to calculate the wetland and stream areas that could be impacted by transportation projects. The road project areas can be represented by a polygon layer and compared to the wetland & channel maps that were derived from previous steps. A simple spatial analysis will show the impacted areas.

However, if the project polygons are not clearly defined, we will predict the impacts based on other information. In the next example, we will use a polyline project layer *Next 4 Fiscal Years Lines* (<https://gis.dot.state.oh.us/tims/Data/Download>) to illustrate this analysis.

The *Next 4 Fiscal Years Lines* layer lacks the necessary information about number of lanes and functional class, so that we cannot calculate the exact width of the new project and cannot create a buffer region

around the road centerline. Therefore, we assume an average buffer width of 50 m, and calculate the wetland areas within the 50 m distance.

Take the road in Figure 64 for example. The road can be arbitrarily long in a GIS layer, so it is reasonable to calculate the local impacts instead of the total impacts. The first step is to break down a polyline to multiple vertices. However, the vertices are usually not evenly spaced (Figure 64).



Figure 64. A future project road over an aerial imagery.

To make the points evenly spaced, use the Densify tool (Figure 65, <https://desktop.arcgis.com/en/arcmap/latest/tools/editing-toolbox/densify.htm>). The tool modifies the original input layer. Choose DISTANCE and choose a proper distance. We chose 5 m in this case. Usually, there is no need to choose a smaller distance, e.g., 1 m.

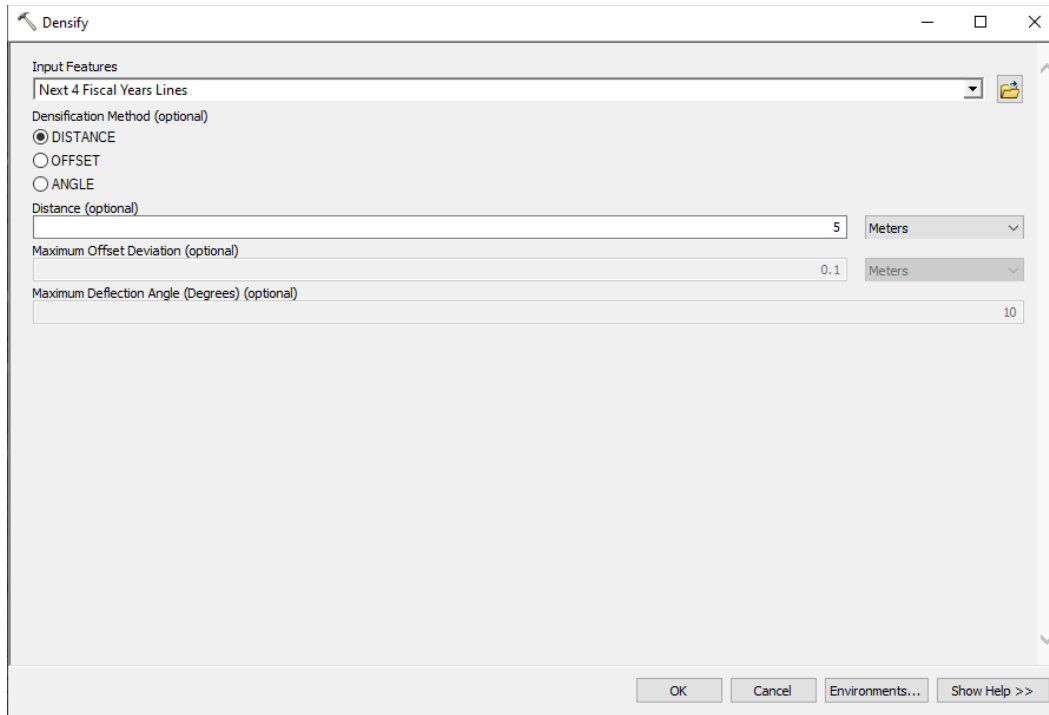


Figure 65. Use the densify tool to increase the density of vertices.

After applying the densify tool, the number of vertices substantially increased and their spacing is even (Figure 66). Next, convert the polyline layer to a point layer using the Feature Vertices To Points tool (<https://desktop.arcgis.com/en/arcmap/latest/tools/data-management-toolbox/feature-vertices-to-points.htm>). Figure 67 is the result of this conversion.



Figure 66. Vertices after applying the densify tool.



Figure 67. Converting the polyline to points.

We will create buffer regions from the points. Use 50 m as the buffer distance, and the buffer results are shown in Figure 68. Those are circles because of the input features are points.

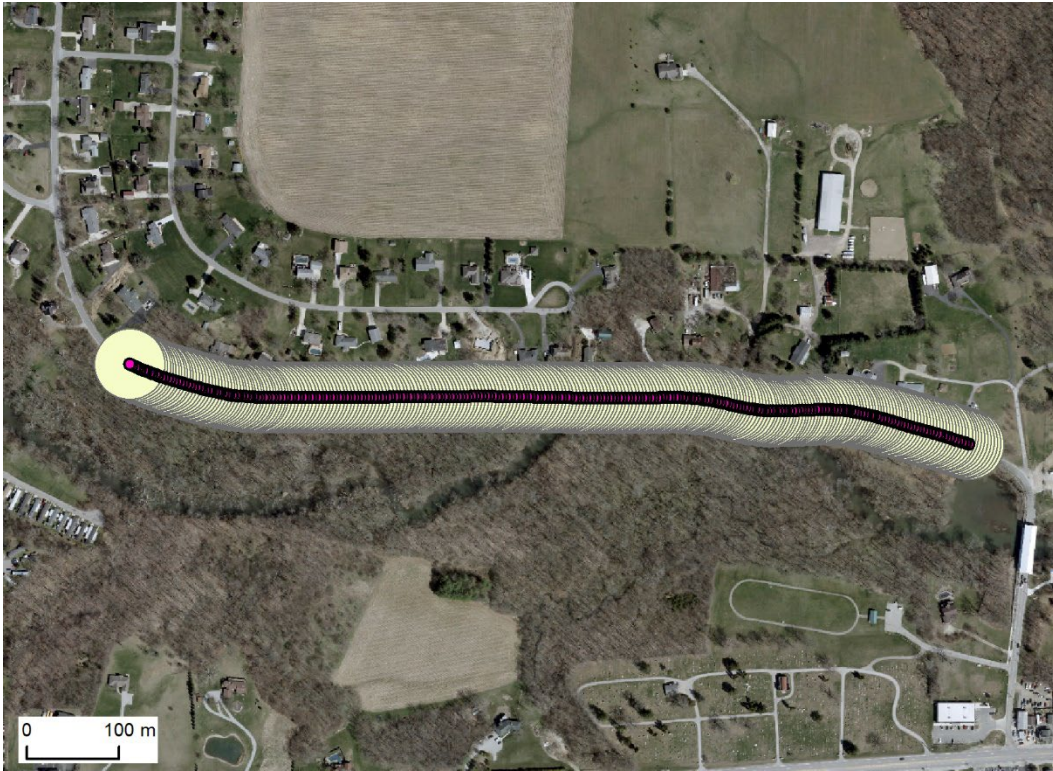


Figure 68. 50 m buffer regions of the points.

Note the circles are overlapping. If we use them to calculate the areal impacts, ArcGIS may give errors. This has been a bug in ArcGIS for a while. We will split the circles to multiple layers so that no two circles in one layer can overlap. It is impractical to create layers with only one feature, though it is guaranteed to have no overlap. Instead, we use the mathematical modulo (remainder number of the division) to solve this problem. Open the attribute table of the circle layer and create an integer field named Mod (Figure 69). Calculate the modulo with the equation in Figure 69. Note that we are calculating the modulo of the ID field divided by 25. The ID is a unique identification and can exceed one million (the exact reason to avoid creating single feature shapefiles), however, the modulo by 25 can have maximum 25 possibilities, from 0 to 24.

Why did we choose the number of 25? Because the radius is 50 m, and two non-overlapping circles should be spaced at 100 m away. The point spacing is 5 m, so that every 20 (= 100 m / 5 m) points should not overlap each other. Therefore, we chose 25, a number slightly over 20.

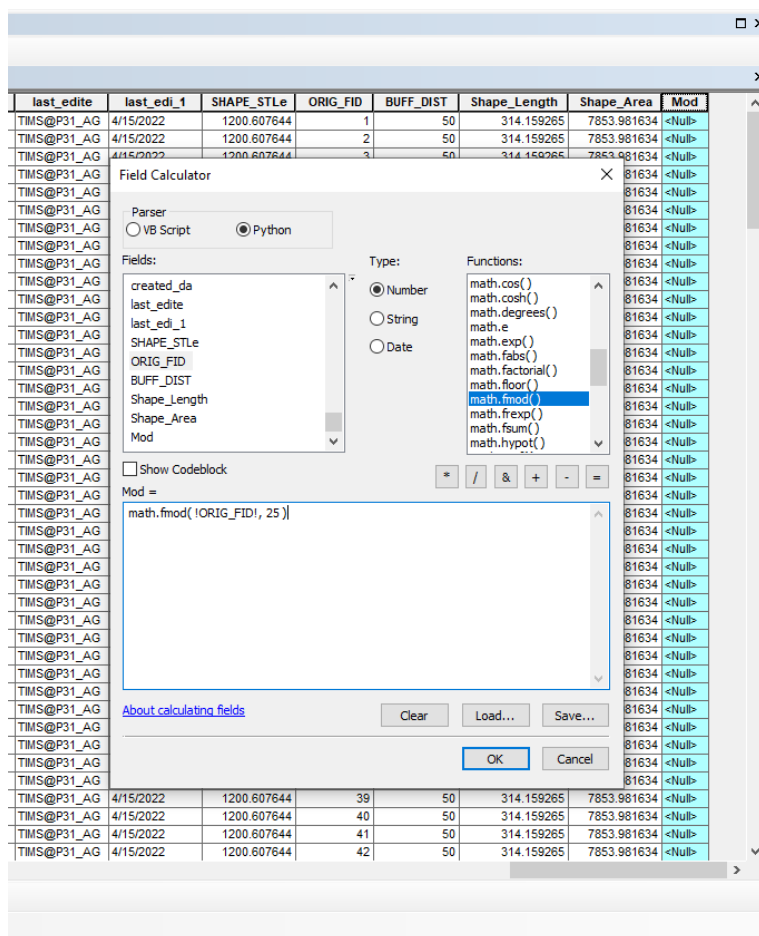


Figure 69. Creating and calculating a mod field.

Next, split the circle layer with the Split By Attributes tool

(<https://desktop.arcgis.com/en/arcmap/latest/tools/analysis-toolbox/split-by-attributes.htm>).

Remember to choose the *Mod* field for the Split_*Fields* parameter input. Figure 70 shows one of the split layers and not any two circles overlap.



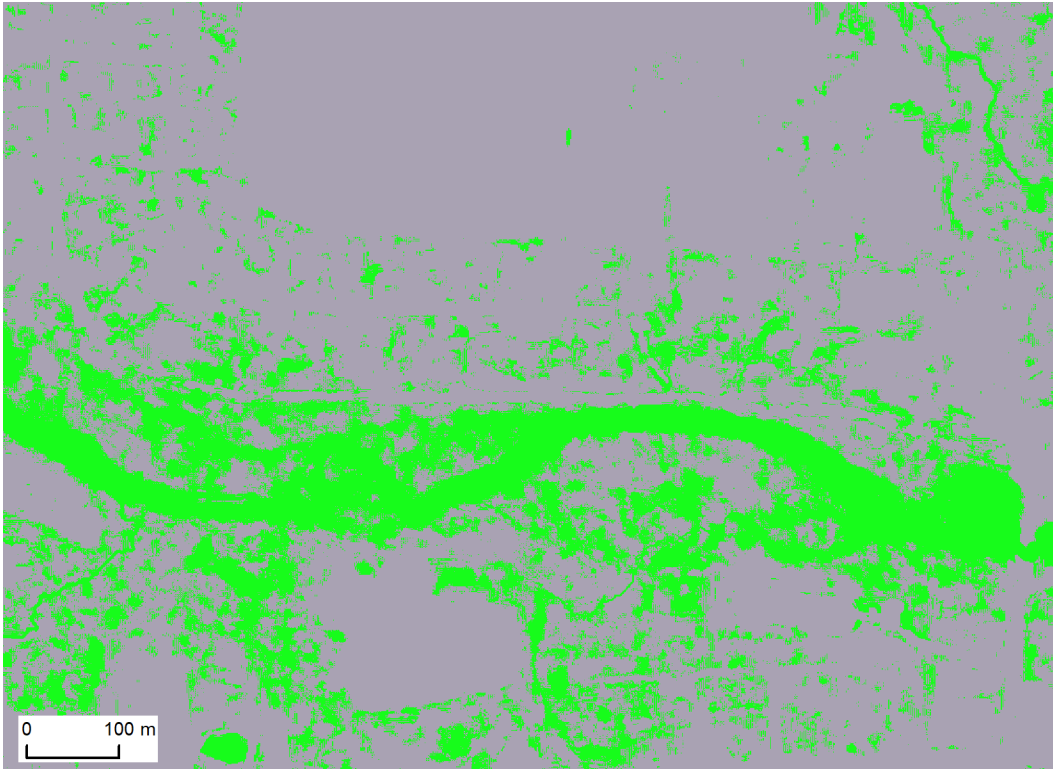
Figure 70. Creating and calculating a mod field.

Next, we will prepare a binary wetland layer based on the previous mapping results. For image classification, reclassify the result map so that wetland classes all have the value of 1, and upland classes have the value of 0. For channels, assign 1 to channel and 0 to non-channel. Then, combine the wetlands and channels to one binary raster layer using the rules below.

$$\text{Raster value} = \begin{cases} 1, & \text{if pixel is wetland or channel} \\ 0, & \text{otherwise.} \end{cases}$$

The result is shown in Figure 71 where green is wetland (=1) and gray is upland (=0).

Next, use a custom Python script to calculate the wetland impact (Figure 72). The impact is a floating number from 0 to 1, representing the ratio of impacted wetland area within each circle. The script will calculate the impacts of each layer and merge the results to a single table. Make necessary changes to variable names and file directories.



```
Zonal Overlapping Polygons - Copy.py - E:\OneDrive\Python Tools\Zonal Overlapping Polygons - Copy.py (2.7.18)
File Edit Format Run Options Window Help
import os
import arcpy
from arcpy import env
from arcpy.sa import *
env.workspace = 'C:/temp/'
env.overwriteOutput = True
arcpy.CheckOutExtension("Spatial")
arcpy.management.CreateFileGDB('C:/temp/', 'tables.gdb')

for fc in os.listdir('C:/temp/'):
    if fc.endswith('.shp'):
        ID = fc.replace('.shp', '')
        ZonalStatisticsAsTable(fc, 'ORIG_FID', 'C:/temp/wetlands.tif', 'C:/temp/tables.gdb/t'+ID, 'DATA', 'MEAN')
        print fc
        arcpy.Delete_management(fc)
print 'done'
```

Ln: 16 Col: 12

The final step is to join the table back to the point layer (Figure 73). Here the values mean the ratio of impacted wetland area over the entire circle area, from no impact (0) to total impact (1).

