Raster Analysis with arcpy

Overview

1. Raster Data Structures
2. arcpy Raster Class
3. ArcGIS Raster Quirks
4. Spatial Analyst Module
5. Spatial Analyst Map Algebra

Raster Data Structures

Raster Data Structures

- Raster data is made up of rows and columns
- Coords are typically in (row, col) format, but sometimes (x-offset, y-offset)
- Coord point of reference is the origin in the upper-left
- Raster structure is very much like a 2-dimensional array
    - See numpy

arcpy Raster Class

arcpy Raster Class

- arcpy provides the Raster class to work with raster data in python
- Raster objects are created by calling `arcpy.Raster(path_to_raster)`
- Raster objects are returned from spatial analyst functions/operations

arcpy Raster Class

- Raster objects, if not referencing a dataset on disk, are not persistent
    - Stored in the workspace of any input datasets used in their creation, or
    - Stored in the scratch workspace
    - Not contained in RAM
- To keep a raster result, use the `Raster.save(path_to_raster)` method

arcpy Raster Class

- Raster objects have many properties
  - Can use `arcpy.Describe()`, but likely what you need to know can be gotten directly from the raster object
  - Use `arcpy.Describe()` when you do not want to open a raster dataset as an object (not sure if there is an advantage either way), or need properties not provided by the Raster class

ArcGIS Raster Quirks

ArcGIS Raster Quirks

- As mentioned, the underlying data behind Raster objects is stored on disk
- But where is this data stored on disk?

ArcGIS Raster Quirks

- Created raster data will likely be in one of two places:
  - The workspace of one of the input dataset
  - The scratch workspace as defined in the env settings
- Why do we care? Will we not save any data we care about keeping to a specific and known location?
  - It turns out that some raster formats have *significant* limitations
  - Moreover, where the data is stored determines the format
  - **This is a big deal**

ArcGIS Raster Quirks

- Raster data stored in a folder workspace by ArcGIS is, by default, in ESRI GRID format
    - Terrible data format: has errors if
        - file name is too long
        - a directory name in the path is too long
        - the path contains spaces
        - etc.
    - We *cannot* change the format for temporary raster data created by the Spatial Analyst tools in arcpy
    - We *can* setup a scratch folder we know works
        - Make a folder `C:\scratch` or something and set it as the `env.scratchWorkspace`

ArcGIS Raster Quirks

- Raster data stored in a file geodatabase is in a format specific to FGDBs
  - Better data format: does not have limitations of GRID
  - Can create a file geodatabase anywhere in the directory tree and set it as the scratch workspace
  - Input data from a geodatabase will use that geodatabase as the scratch workspace for any derived raster data by default
- **This is a real problem:** Exercise 2 data in geodatabase format to avoid processing errors

Spatial Analyst Module

Spatial Analyst Module

- The spatial analyst extension for ArcGIS provides many raster analysis capabilites
    - Surface tools (e.g., terrain processing)
    - Hydrology tools (e.g., stream delineation and flow)
    - Density tools (e.g., density mapping)
    - Overlay tools (e.g., multi-criteria evaluation)
    - Raster math, conditional, boolean, and map algebra tools
    - etc.

Spatial Analyst Module

- The spatial analyst module contains many classes for specific operations

Spatial Analyst Module

- The spatial analyst module contains one python-specific function, `arcpy.sa.ApplyEnvironment`
    - Creates a copy of a raster as a Raster object using any environment settings
        - Temporary raster clip
        - In-memory raster project
        - etc.

Spatial Analyst Module

- Separately-licensed extensions (like Spatial Analyst) need to be "checked-out" in arcpy before use
- "Checked-out" licenses need to be "checked-in" when no longer needed to ensure they are available for other processes/users
- arcpy provides specific functions for this license-getting process

Spatial Analyst Map Algebra

Spatial Analyst Map Algebra

- Map algebra is essentailly raster math:
  - c = a + b, where the cells in c would be the cells in a plus the cells in b
  - Map algebra also supports boolean operators and comparisons
- ArcGIS offers the Raster Calculator tool for map algebra
  - It turns out expressions entered into the raster calculator are python!
  - In other words, **do not** use the raster calculator in python
  - Note: if a raster calculator operation does not use an operator but has a function-like syntax (Con, SetNull), then what it is really doing is calling a toolbox tool. You can do the same in python, using something like `arcpy.sa.Con()` or `arcpy.sa.SetNull()`.

Spatial Analyst Map Algebra

An example of raster algebra in arcpy:

```
raster1 = arcpy.Raster(r"C:\data\raster.img")
raster2 = arcpy.Raster(r"C:\data\geodatabase.gdb\stuff")

suitable = raster1 >= 357 | (raster1 < 53 & raster1 >= 14)

raster2 = (raster2 ** 2) / 13.5

suitable &= raster2 == 4 | raster2 == 19

suitable.save(r"C:\data\geodatabase.gdb\suitable")
```