GEOG 432/832: Programming, Scripting, and Automation for GIS

Unit 08.02: Raster math and reclassifcation

Dr. Bitterman

Today's schedule

- Open discussion
- Slides, discussion and exercises
    - Wednesday: raster algebra and more
- For next class

Open discussion

Raster data structure review

1. What's a raster dataset?
2. How are they used?
3. What are the key properties of a raster?
4. Any other concerns (e.g., toppolgy) that you're aware of?

Today's prep:

- We're using the same data as Monday (*week08inclass.zip*)
- Open a new project (or the one from Monday)
- Start a new Jupyter notebook in ArcGIS Pro
- Setup your workspace

Reclassification

- What does it mean to "reclassify" a raster?
- When might you reclassify a raster?
- What does your textbook say about how we do it?
  - Values
  - Ranges
- Using our for loop from Friday, how might we implement a reclassify?

Using built-in methods

Syntax:

```
Reclassify(in_raster, reclass_field, remap, {missing_values})
```

- in_raster: the raster you're working on
- reclass_filed: the field you are reclassifying
- remap: an object that describes the reclassifcation
- {missing_values}: how to handle missing values

From last class:

How did this work again?

```
myraster.save("nlcd_test")
testraster = arcpy.Raster("nlcd_test")
testraster.readOnly = False

for x, y in testraster:
    if testraster[x,y] == 41 or testraster[x,y] == 81:
        testraster[x,y] = 71

testraster.save()
```

Reclassifying with values

How does this one work? Let's break it down

```
myraster = arcpy.Raster("nlcd_lc_14n")
myraster.save("nlcd_test")

testraster = arcpy.Raster("nlcd_test")
testraster.readOnly = False

myfield = "VALUE"
myremap = arcpy.sa.RemapValue([[41, 71],[81, 71]])

outraster_value = arcpy.sa.Reclassify(testraster, myfield, myremap)
```

What happened?

Reclassifying with ranges

break it down

```
myraster = arcpy.Raster("nlcd_lc_14n")
myraster.save("nlcd_test")

testraster = arcpy.Raster("nlcd_test")
testraster.readOnly = False

myfield = "VALUE"
myremap = arcpy.sa.RemapRange([[41, 51, 71], [80, 89, 71]])

outraster_range = arcpy.sa.Reclassify(testraster, myfield, myremap)
```

How would you test/display what cells changed?

A simple change detection

What's this code doing?

```
change_detect_val = myraster == outraster_value
```

Raster algebra

Arithmetic functions:

+, -, /, *

Relational functions:

==, <, >, !=

Also: Boolean and bitwise (see your book)

Assumptions of raster algebra:

- Same resolution
- Same extent
- Orthogonal

(But Arc will "cheat it" for you)

You have some rasters… give it a shot. Try out a few operators

Your final task:

- Imagine a habitat suitability analysis for the will spotted hookbilled two-legged platypus (it's real, trust me ;))
- Criteria: only lives in wetlands below 400m
- Your task: find the suitable habitat using the tools we used today

(do we want to start on the whiteboard?)

For next class

- Lab 4 starts Friday
- Next week's readings are linked on Canvas