Preparing Criteria Classes for the Representation Analysis

Created: January 23, 2014

This document provides step-by-step instructions for preparing the conservation features for the representation analysis using ArcPY geoprocessing tools. The script can either work using Python IDLE or the Python editor for Windows PythonWin. To load your script into Python IDLE, right click on the .py script and select “Edit using IDLE”. Make sure the parameters setting is properly filled (see instruction below). When the parameters have been filled and the script is ready to be launched, go to the Run menu and select Run Module command.

PythonWIN is a Windows IDE and GUI framework for Python. It has an [integrated debugger](http://docs.activestate.com/activepython/2.5/pywin32/html/pythonwin/doc/debugger/index.html) and a rich Python editing environment. PythonWin can be downloaded from <http://starship.python.net/crew/mhammond/win32/Downloads.html>. A shortcut can be created during the installation process to add “Edit using PythonWin” in the right clic menu. The same attention should be put on setting the parameters properly before launching the script.

## SETTING UP PARAMETERS

**Step 1 - Setting up project directories**

The first step in the preparation of the criteria that will serve in the representation analysis is to create a project directory along with sub-directories that will contain input data and the Python script. Note that the output directory and sub-directories are created by the script.

- StudyRegionName

- catchments

- criteria

The “catchments” folder contains the catchments shapefiles within the study region. The“criteria” folder contains the criteria of interest in TIFF format. They can be continuous or categorical grids. Criteria don’t have to be clipped at the study region level. Both catchments and criteria used as input should use the same spatial projection.

**Step 2 – Specifying the location of the catchments and criteria**

The second step is to specify the path where the script is and derived i) the location of the catchments shapefile and, ii) the location of all criteria needed in the representation analysis from it. To derive the catchments and criteria paths, the script should be stored in the StudyRegionName folder at the same level of the catchments and criteria folders.

# Extract the working directory defined by the location of the script.

currentDir = os.getcwd()

# Location of catchments shp

catchments = currentDir + "catchments/catch.shp"

# Location of the criteria in TIFF format

LED = currentDir + "/indicators/led.tif"

CMI = currentDir + "/indicators/cmi.tif"

GPP = currentDir + "/indicators/gpp.tif"

NALC = currentDir + "/indicators/nalc.tif"

**Step 3 – Specifying the name of a temporary gdb and the catchment unique identifier.**

A gdb has more flexible properties and settings than a shapefile. It allows to control more parameters which is the reason why we use in the intermediate step between the input and output file. A gdb output name and the unique catchment identifier from the catchments input table need to be defined prior to run the script.

# Name of the temporary gdb used to generate the final result

indicatorsGDB = "indicators.gdb"

# Set the unique catchment identifier used in the catchments shp

catchID = "CATCHNUM"

**Step 4 – Specifying the list of criteria used.**

The representation analysis uses both continuous and categorical criteria. The preparation of the 2 types of data is not process the same way. Continuous criteria are reclassified according to a defined number of classes and methods prior to be used in the representation analysis, while categorical criteria are not reclassified. To allow the script to operate the proper operation on the criteria, the user needs to provide 2 set of list according to their nature. Both lists store the path of the respective criteria.

# List continuous criteria using the input path.

conIndicatorList = [CMI, GPP, LED]

# List categorical criteria using the input path.

catIndicatorList = [NALC]

**Step 5 – Setting up output names.**

Another setting is to specify the criteria output names. Output names are stored into a dictionary structure (inside the quote). It allows to keep information on the source, years, etc. For example, the output cmi criteria could be named “cmi\_strlbg1975”.

# List continuous indicators output name

conIndicatorName = {CMI:"CMI", GPP:"GPP", LED:"LED"}

# List categorical indicators output name

catIndicatorName = {NALC:"NALC"}

**Step 6 – Specifying slice parameters.**

The reclassification is done using Spatial Analyst/Slice tool from ArcGIS. The Slice tool doesn’t work well with floating raster type. Thus a list of floating raster needs to be provided to allow the script to operate a transformation before processing the Slice tool. The list should store the criteria output name specified within the quote in step 5.

# List all criteria that uses raster pixel FLOAT type

floatRast = ["LED"]

The Slice tool also needs to have the reclassification method and number of classes specified. The methods allowed are EQUAL\_INTERVAL, EQUAL\_AREA and NATURAL\_BREAKS.

# Set the reclass methods proposed by the Slice tool in ArcGIS

method = "EQUAL\_INTERVAL"

# Set the number of class to process the reclassification

n = 20

## GENERATING REPRESENTATION ANALYSIS FUNCTIONS

ALL FUNCTION USED IN THIS SCRIPT ARE HARDCODED. THE PARAMETERS ARE TAKEN FROM THE PARAMETERS SETTING SECTION SPECIFIED IN STEP 1 TO 6. NOTHING SHOULD BE CHANGED IN THE FUNCTIONS WHILE RUNNING THE SCRIPT.

**Step 7 – Preparing catchments**

In step 7, the prepCatch function creates a copy of the original catchments shp in the gdb. It allows not to work with the original dataset and reduce the risk to corrupt the data by doing unintended operation. The function also deletes all unnecessary fields except the unique catchment identifier.

outCatch = tempOutputDir + "/" + indicatorsGDB + "/catchIndicators\_" + str(n) + "\_" + method

prepCatch(catchments,tempOutputDir,indicatorsGDB,n,method,outCatch)

**Step 8 – Reclassifying continuous criterion**

In step 8, the reclassConCriteria function i) loops through a list of continuous criteria, ii) reclassified them according to a method and number of classes specified in step 6 and, iii) cross-tabulate classes area with a set of catchments using the unique catchment identifier as processing unit. All classes areas per catchment are expressed in squared meters.

reclassConCriteria(outCatch,catchID,conIndicatorList,conIndicatorName,floatRast,tempOutputDir,n, method)

**Step 9 – Generating reclassified criterion SHP**

In step 9, the genConCriteriaSHP function standardizes the output based on the number of classes specified in step 6. Because the criteria are usually generated at the Area of Assessment level and the study region is often represented by a set of catchments at the ecoregion level, some criteria classes may be absent and won’t be computed in step 8. To make sure all classes will be represented in the final output table, the genConCriteriaSHP i) creates the columns based on the number of classes and, ii) copy the result obtained in step 8 in a new output table. All classes that are absent in the ecoregion will result with a tabulate area of 0 for all catchments. The name of the output table is specified using the name of the indicator, the number of classes used to reclassify the input and the reclassified methods. The output file will be used in subsequent steps.

genConCriteriaSHP(outCatch,catchID,conIndicatorList,conIndicatorName,tempOutputDir,FinalOutputDir,n,method)

**Step 10 – Tabulating categorical criterion per catchment**

In step 10, the tabulateCatCriteria function generates a table where each categorical classes have their area compiled per catchment. The name of the output tables are specified using the name of the criteria. The output file will be used in subsequent steps.

CatCriteria(outCatch,catchID,catIndicatorList,catIndicatorName,tempOutputDir)

**Step 11 – Generating categorical criterion SHP**

In step 10, the genCatCriteriaSHP function standardizes the output based on the number of classes present in the original categorical grids (see explanation in step 9). The output file will be used in subsequent steps.

genCatCriteriaSHP(outCatch,catchID,catIndicatorList,catIndicatorName,tempOutputDir,FinalOutputDir)

**Step 12 – Generating representation analysis input table**

In step 11, genDBFCriteria generates a list of catchments and assembles all standardized criteria output tables from step 9 and 11 to the table. The output file is then saved as a dbf that can later be openned using Excel.

genDBFCriteria(outCatch,FinalOutputDir,catchID,n,method)