**Refining an Automated Model for Basic Landform Classification**

**A Python and ArcGIS Approach**

Josh Moss

Penn State MGIS

Capstone Report

Patrick Kennelly, Advisor

**Abstract**

A rigorously classified landform spatial data layer based on the science of geomorphology is useful as a broad mapping guideline for a variety of land use and environmental planning applications. Existing automated models based on digital elevation models (DEMs) are not available as tools compatible with ArcGIS and they introduce errors into the classified landform layer. This paper presents an improved method of classifying landforms from DEMs along with a toolbox file for ArcGIS that is optimized for functionality based on an original Python script. The improved model is verified by comparison with landform classification conducted through photographic interpretation.

**Introduction**

The ability to classify and delineate landforms is important to many different applications of geographical information science (GIS), particularly the field of geomorphology and for use in any analyses that wish to incorporate a geomorphological understanding into their approach by analyzing other types of spatial data using geomorphological landform units as a variable for comparison (Minar and Evans 2008). A rigorous geomorphological landform layer against which to perform analyses could be useful in many fields, including biology, ecology and forestry, archaeology, and lands management (MacMillan 2004). Landform classification is also useful in specific efforts such as landslide susceptibility mapping (Dikau 1990, Pike 1988), precision agriculture (Klingseisen et al 2007), and soil condition modeling (Schmidt 2004). Landform classifications, however, unlike hydrological, elevation, soil or photographic data, cannot be obtained through mechanical means such as LiDAR, radar, ground testing or photography. They must be derived from manipulations of available data based on geomorphological theory and understanding.

Traditional methods of defining landforms involved a great degree of subjectivity. In the past, soil surveyors and geomorphologists had to interpret photographs visually using established geomorphological standards to assign landform classifications to the landscape (Klingseisen et al 2007). A leader in this field was geomorphologist Edwin Hammond, who developed a mathematically based system for classifying landforms on a broad scale based on three parameters: slope, relief and profile. Slope can be defined as vertical inclination, relief as the vertical dimension, expressed as the range of difference in elevation in a limited area. Profile, for Hammond’s purposes, is defined as an index of the relative percentage of land in each slope class (e.g. gentle [less than 8%]) falling into the top or bottom half of the elevation range for the area in question. These three mathematical attributes allowed Hammond to derive 96 possible landform classes, 45 of which he determined occurred commonly across the continental United States (Hammond 1954). Hammond’s system was used with success world-wide. With the advent of GIS, there have been many efforts to automate Hammond’s approach or similar techniques (Minar and Evans 2008; Morgan and Lesh 2005; Dikau et al 1990). In the 1990’s, geomorphologist Richard Dikau, working with the U.S. Department of the Interior, adopted and generalized Hammond’s method for use with early GIS using Digital Elevation Model (DEM) data as the input. Morgan and Lesh’s 2005 paper presented a series of geoprocessing steps for Esri’s ArcGIS software that would approximate Dikau’s adaptation of Hammond’s system, along with steps for a more generalized system of landform classification. The more generalized process is still based on Hammond’s original work, with additional guidance by the Missouri Resource Assessment Partnership’s (MORAP) landform classification procedure (Morgan and Lesh 2005). The purpose of this paper is to use the steps suggested in Morgan and Lesh’s MORAP landform classification method to create and validate a custom tool for ArcGIS to allow users to generate classified landform layers with minimal input or background knowledge.

**Methods**

Five 100 square meter study areas were chosen (Figure 1) to demonstrate the fitness of the model over a wide variety of topography. Study areas were selected by some of the earliest work in macro-level geomorphology, the physiographic regions of North America delineated by pioneering geomorphologist Nevin Fenneman (1914). With his work as a guide, study areas were chosen in the Sierra Nevada Mountains, the Louisiana Coastal Lowlands, the Piedmont region of Virginia, the Great Plains of Kansas, and the Northern Great Basin, along the border of Nevada and Oregon. This provided a wide range of topography, including the highest and lowest points in the Continental US, canyons, rolling hills and featureless flats, against which to test the fitness and accuracy of the model as originally laid out for ArcGIS software by Morgan and Lesh (2005).



Figure 1: Five study areas representing diverse topography, based on the physiographic regions defined by early geomorphologist Nevin Fenneman (1914).

Initial testing showed that there were some flaws in the geoprocessing steps laid out by Morgan and Lesh (2005) in handling certain types of topography. The outputs were generally within the expected classes, but a flaw was identified where the input DEM contained negative integers. Negative integers in a DEM represent areas that lie below sea level, which were present in the Sierra Nevada study area, which encompassed the low-lying Badwater Basin area of Death Valley, and in the Coastal Lowlands study area. These areas produced anomalous results (Figure 2), indicating that various steps in the procedure were not tuned to deal with negative integers.

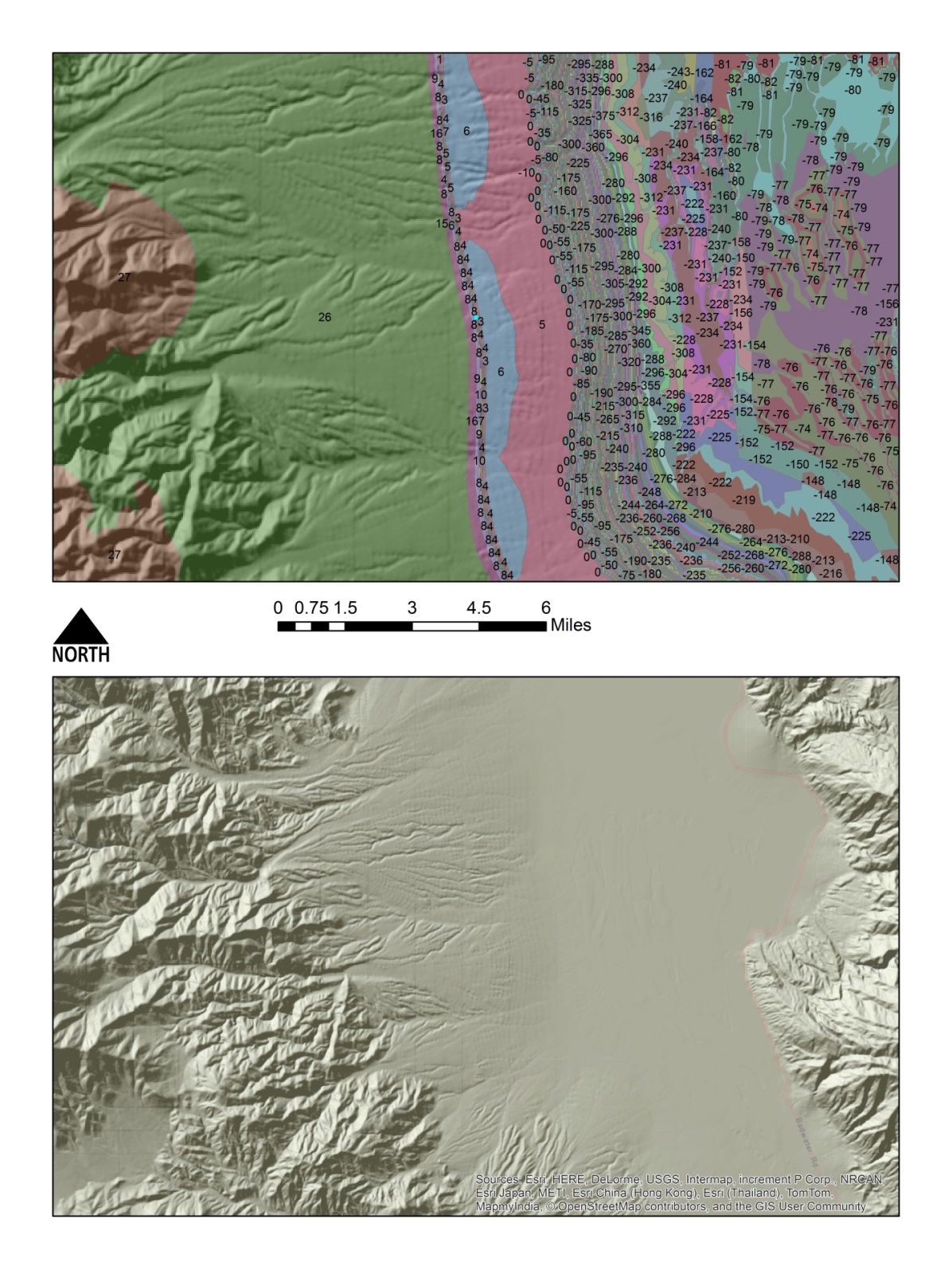


Figure 2: The image above shows a hillshade rendering of the Badwater Basin with and without the classified landform overlay, showing where the classification procedure generated many anomalous results in areas lying below sea level. Some anomalous codes, such as the values in the range of 0 to 6 in the center of the map or -75 to -81 on the eastern edge, seem to correspond with actual elevation. Others, in the negative triple digits, are more unexpected.

The other major error observed during testing was a failure of the model to distinguish between concave and convex areas of extreme relief. This error was spotted in the Great Basin Study area, which is cut by many deeply incised canyons. These areas were classified with codes within the expected range; however, these canyon areas were consistently coded as hills or mountains (Figure 3). The model had been able to observe the relief, but not the convexity. This error seemed to have potential to impact the results for many of the possible use cases for the model, especially a use such as landslide susceptibility mapping (Dikau 1990).

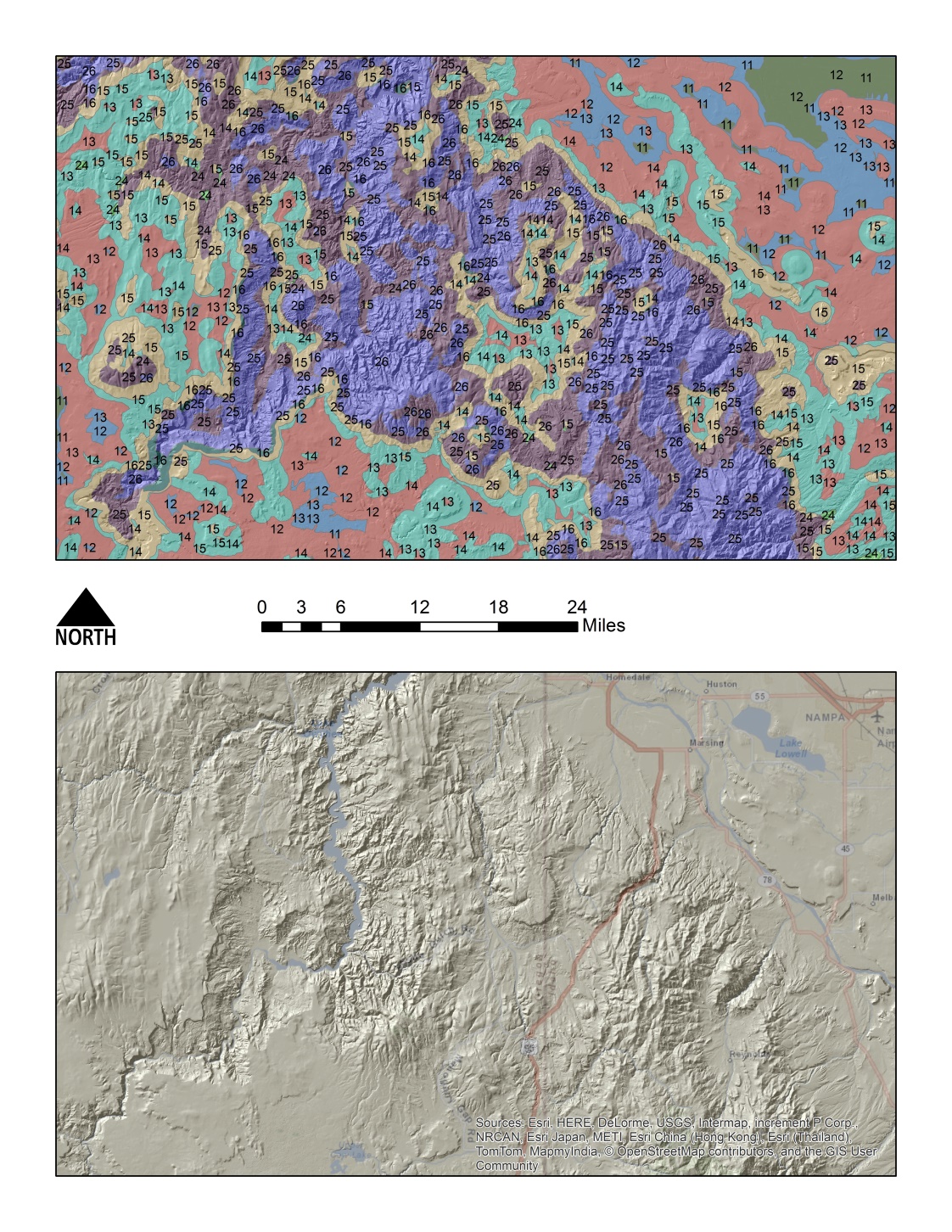


Figure 3: When comparing the classified landform overlay and hillshade layer above that show a part of the Great Basin study area, it is clear that the model interpreted the concave relief of the canyon in the eastern portion of the map as convex. The codes 24 (brown), 25 (purple) and 26 (blue) represent low hills, hills and mountains, respectively. Water can be observed in the canyon bottom on the lower map, where the original 2005 Morgan and Lesh model coded hills and mountains. Without the hillshade layer as a reference, the canyon system that cuts from the center of the map to the southwest would be indistinguishable from the hills in the south eastern portion of the map by the codes alone.

**Error Mitigation**

Both errors found during the testing phase in the model as proposed by Morgan and Lesh (2005) were mitigated by adding geoprocessing steps to the model (Figure 4). The negative integer error was solved by introducing a Map Algebra step that adds a value of 100 meters to the study area DEM, bringing all values above zero while maintaining relative elevation relationships throughout the study area. The value of 100 meters is suitable for classifying landforms within the United States as the lowest point in the contiguous US is the Badwater Basin in Death Valley at 86 meters below sea level. This value might need to be altered within the model in order to classify landforms on other continents.

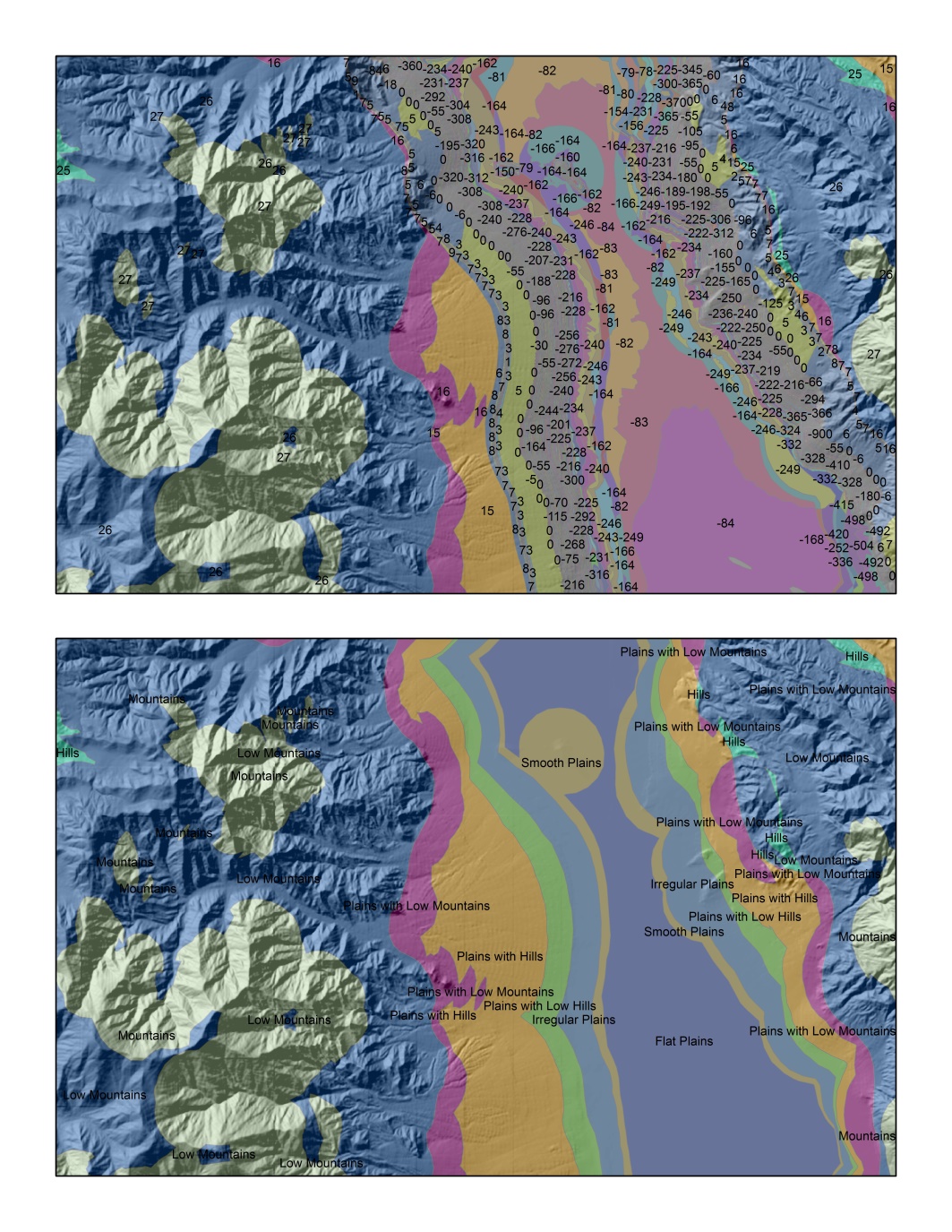


Figure 4: The image above shows the Badwater Basin area with the classification overlay from the model before (top) and after (bottom) the error mitigation step for the negative integer error was introduced. The top image shows the numerous anomalous codes generated for areas below sea level by the original model when negative integer values were in the input. No anomalous codes are present in the bottom image, allowing it to be labeled with the corresponding landform names.

The conflation by the model of areas of extreme convex and concave relief was solved by adding a measure of elevation relative to the study area to the classification procedure (Figure 5). This was achieved by reclassifying the study area DEM into “elevation zones” and producing a raster layer that selected the lowest parts of the study area, the lowest two “elevation zones” to be cross-calculated with the output of the model to produce additional classification values that represent low elevation areas of high to extreme concave relief. Experimentation resulted in three additional classes, “arroyo”, “canyon” and “steep canyon”, mimicking the convention from the original model of having two to three gradations in relief or sub-types for each landform type. This procedure was coded into the new version of the model as an option that can be toggled on or off, due to the fact that if canyons are not present in the study area, it will cause misclassification of low-lying areas. For this reason, the use of this option requires some a priori knowledge of the study area on the part of the user. Additionally, the elevation values used to reclassify the DEM into the “elevation zones” necessary for this procedure are tuned for areas roughly 1,600 to 1,800 meters above sea level, which is broadly representative of canyon lands in the United States but might need to be adjusted for other countries.

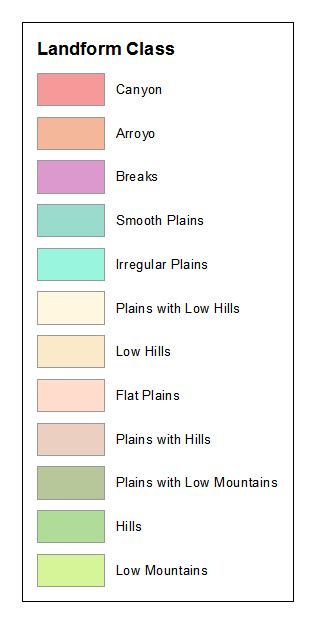
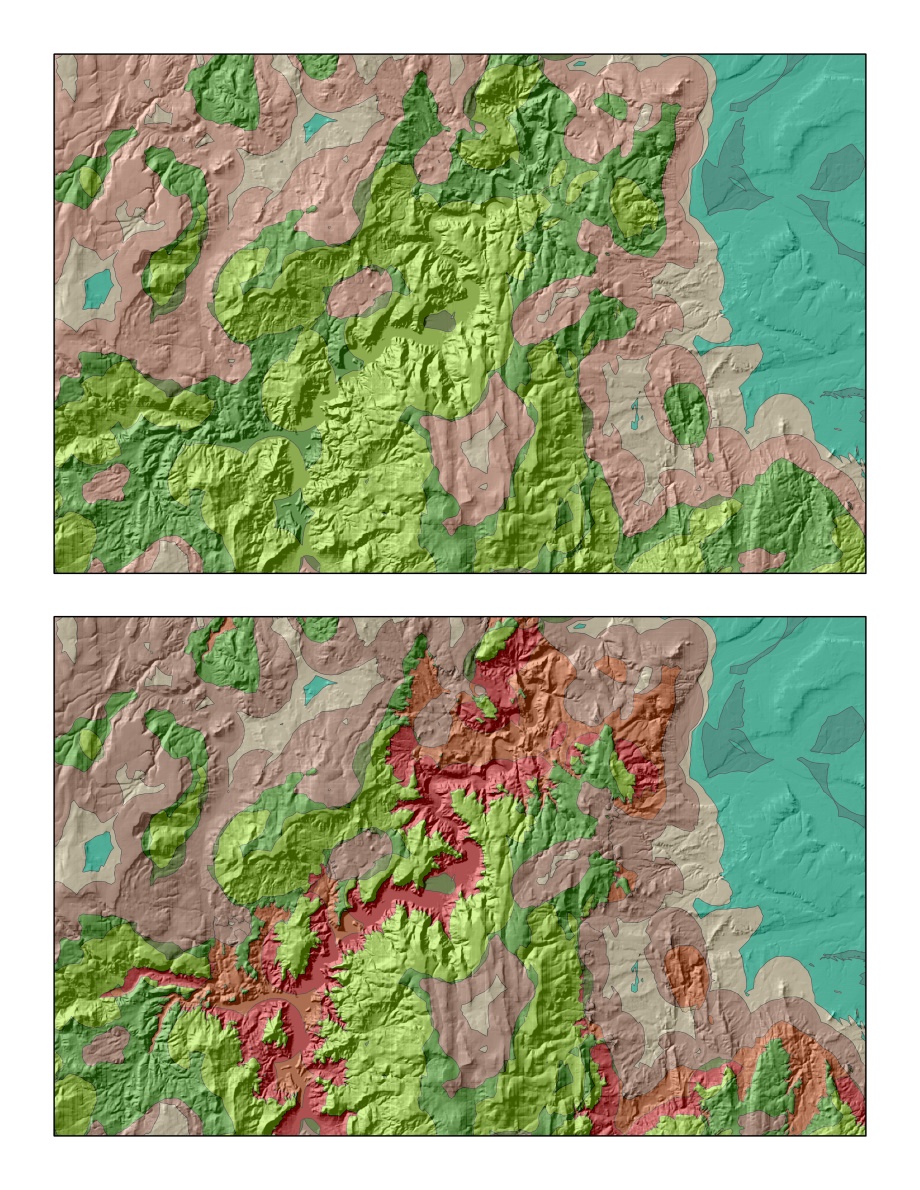


Figure 5: The above image shows the same canyon depicted in Figure 3 classified with (bottom) and without (top) the canyon option. The bottom image shows how the canyon option calculations were able to identify and classify areas of moderate to extreme concave relief. Areas falling into the “canyon” and arroyo class were identified. No areas were identified for either of the convex or concave classes representing the most extreme relief (“mountains” and “steep canyons” were present in this study area.

**Model Production**

The improved model was produced using Python in the PythonWin integrated development environment as a custom tool for ArcGIS with a fully annotated interface. It requires the input of a 30 meter resolution DEM file and outputs a classified landform polygon layer with both numerical landform codes and text-based landform names in the attribute table. The tool is called “LFMapper” and is available for download as an ArcGIS toolbox (.tbx) file, along with the source code, at lfmapper.blogspot.com. The source code is included in this paper as Appendix A.

**Model Verification**

The completed model was verified against a heads-up landform classification across each of the five study areas (Figure 6) and the results were analyzed in an error matrix table (Table 1). This procedure is common for verifying photo-interpretation and remote sensing results and was suggested for the verification of automated landform classification models by Klingseisen et al (2007). Each study area was mapped at a scale of 1:100,000, using the set of classes suggested by the Morgan and Lesh (2005) model, along with the concave relief classes developed during the error mitigation phase, as a classification guide. ArcGIS hillshade layers, National Aerial Imagery Project (NAIP) data, United States Geological Survey (USGS) topographic maps, Soil Survey Geographic Database (SSURGO) data and National Hydrology Dataset (NHD) datasets were used to aid mapping/classification decisions. The 3D profile functionality of Google Earth was also employed to determine mapping break points.

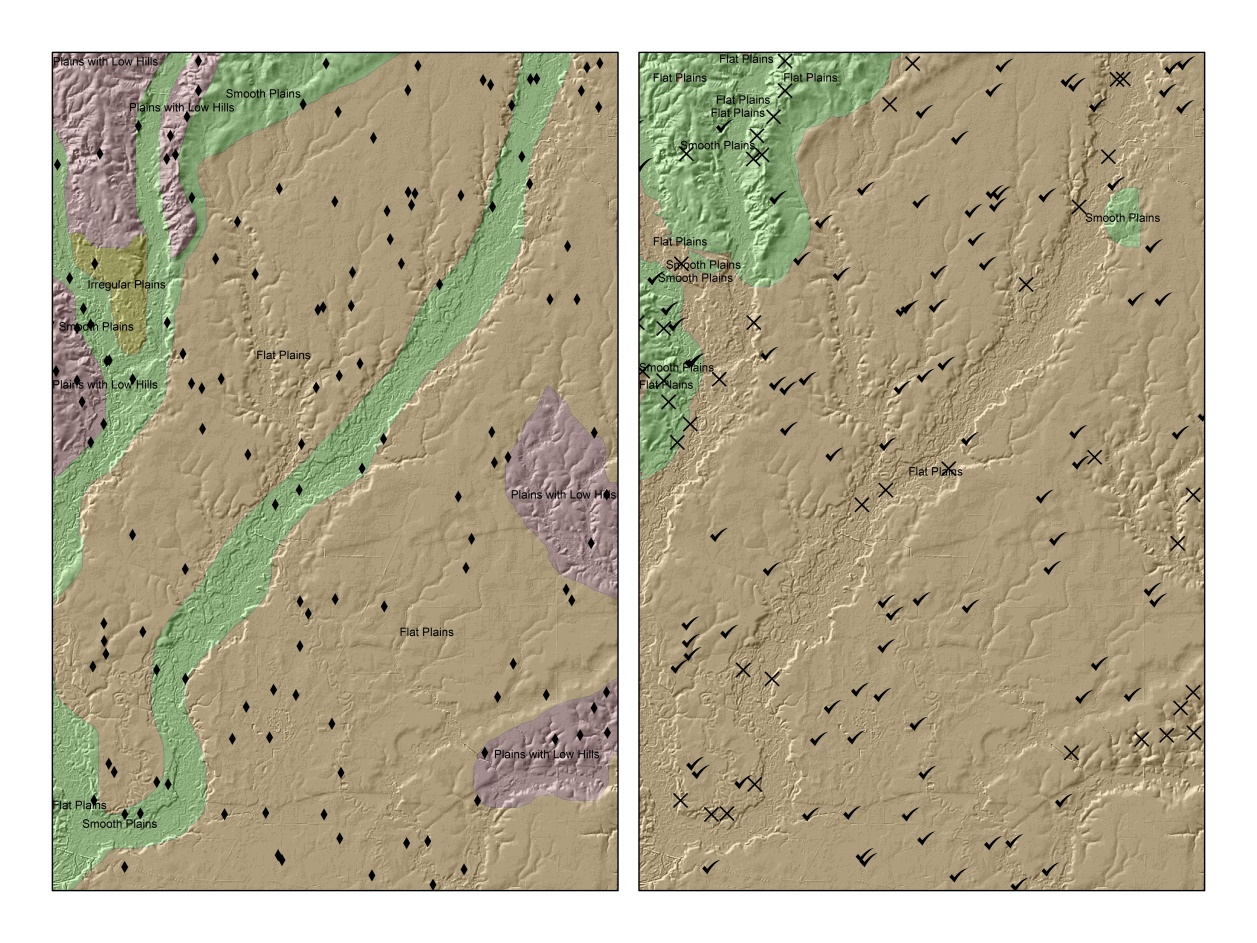


Figure 6: The above image shows a close up of the Coastal Lowlands study area with the error matrix testing points displayed. The map on the left shows the heads-up classification and randomly selected control points, and the map on the right shows the automated classification. The “x’s” on the right represent disagreement between the maps, the checks represent agreement between the classifications.

Each heads-up classified landform polygon layer was then compared to the corresponding model output classified landform polygon layer by generating 1,000 random points per study area to determine the rate of agreement between the two classifications. These results were then analyzed in an error matrix (Table 1).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| LFMapper Heads-up > | **Arroyo** | **Breaks** | **Canyon** | **Flat Plains** | **Hills** | **Irregular Plains** | **Low Hills** | **Low Mountains** | **Mountains** | **Plains With Hills** | **Plains With Low Hills** | **Plains With Low Mountains** | **Plains With Mountains** | **Rough Plains** | **Rugged Plains** | **Smooth Plains** | **TOTAL** | **Percent Correct** |
| **Arroyo** | 0 | 4 | 3 | 0 | 1 | 0 | 10 | 0 | 0 | 8 | 2 | 2 | 0 | 0 | 2 | 2 | 34 | 0 |
| **Breaks** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 1 | 0 | 0 | 0 | 3 | 0 |
| **Canyon** | 0 | 3 | 5 | 0 | 1 | 0 | 10 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 22 | 22% |
| **Flat Plains** | 0 | 0 | 0 | 784 | 0 | 30 | 0 | 0 | 0 | 3 | 73 | 0 | 0 | 0 | 0 | 120 | 1010 | 77% |
| **Hills** | 0 | 3 | 1 | 0 | 45 | 1 | 62 | 9 | 0 | 37 | 13 | 17 | 1 | 6 | 18 | 1 | 214 | 21% |
| **Irregular Plains** | 8 | 0 | 0 | 90 | 0 | 420 | 2 | 0 | 0 | 53 | 303 | 12 | 1 | 75 | 24 | 161 | 1149 | 36% |
| **Low Hills** | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 6 | 2 | 0 | 1 | 0 | 2 | 1 | 13 | 7% |
| **Low Mountains** | 0 | 18 | 0 | 3 | 180 | 19 | 82 | 282 | 95 | 28 | 13 | 14 | 4 | 11 | 21 | 2 | 772 | 36% |
| **Mountains** | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 24 | 45 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 70 | 64% |
| **Plains With Hills** | 1 | 5 | 2 | 27 | 18 | 78 | 28 | 1 | 0 | 69 | 42 | 13 | 3 | 42 | 17 | 21 | 367 | 18% |
| **Plains With Low Hills** | 5 | 1 | 0 | 41 | 0 | 74 | 23 | 0 | 0 | 55 | 56 | 37 | 8 | 58 | 27 | 25 | 410 | 13% |
| **Plains With Low Mountains** | 0 | 3 | 1 | 8 | 3 | 20 | 3 | 3 | 0 | 4 | 2 | 1 | 0 | 7 | 3 | 10 | 68 | 1% |
| **Plains With Mountains** | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| **Rough Plains** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Rugged Plains** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Smooth Plains** | 2 | 0 | 0 | 131 | 0 | 352 | 0 | 0 | 0 | 40 | 61 | 0 | 0 | 7 | 0 | 269 | 862 | 31% |
| **TOTAL** | 16 | 37 | 12 | 1084 | 249 | 996 | 221 | 319 | 140 | 303 | 570 | 96 | 19 | 206 | 116 | 612 |  |  |
| **Percent Correct** | 0 | 0 | 41% | 72% | 18% | 42% | 0 | 88% | 32% | 22% | 10% | 1% | 0 | 0 | 0 | 43% |  |  |

**Results**

Table 1: The above table shows the results of the error assessment for the LFMapper tool against a manual classification of landforms.

The result of the error assessment was 39% rate of overall agreement, which compares favorably to the 43% rate of agreement achieved by Klingseisen et al. (2007) when testing their automated landform classification utility, “LANDFORM” (2007). However, there are other interesting trends to be observed. The numbers highlighted in orange in the error matrix table show high rates of confusion. It is worthwhile to note that the highest rates of confusion took place between classes that would have similar levels of relief, possibly rendering them harder to distinguish in a heads-up classification process. One example is the highest rate of confusion, which is between “smooth planes” and “irregular planes”. Interestingly, both these categories have reasonable percentages correct when compared to the overall rate. This can be explained by the fact that they are both very commonly occurring classes in both classification methods.

The most successful classifications according their by-class percentages correct are classes with notably high or low relief (Table 2). “Flat plains” and “mountains” were the most successfully classified classes by the automated procedure, “flat plains and “low mountains” had the highest rate of success for the heads-up method.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **STUDY AREA** | **PHYSIOGRAPHIC REGION** | **PERCENT AGREEMENT** | **HEADS UP POLYGON COUNT** | **AUTOMATED POLYGON COUNT** | **HEADS UP LANDFORM CLASSES REPRESENTED** | **AUTOMATED LANDFORM CLASSES REPRESENTED** |
| VIRGINIA | PIEDMONT | 22% | 153 | 1218 | 11 | 10 |
| CALIFORNIA | SIERRA NEVADA MOUNTAINS | 28% | 163 | 1126 | 13 | 11 |
| KANSAS | GREAT PLAINS | 51% | 33 | 882 | 3 | 3 |
| LOUISIANA | COASTAL LOWLANDS | 73% | 34 | 158 | 5 | 3 |
| OREGON | BASIN AND RANGE | 21% | 157 | 2951 | 12 | 12 |
| COMBINED STUDY AREAS | ALL OF THE ABOVE | 39% | 540 | 6335 | 16 | 14 |

Table 2: The above table shows a breakdown of the percentage correct for the automated method by study area, along with totals for raw polygon and class count for each method. These values are also displayed for all study areas combined.

Table 2 provides a different view of the fitness of the model and the situations in which it performs best. The highest percent correct is in the Coastal Lowlands study area, which has the most homogeneous topography of any of the study areas, followed closely by the Great Plains study area, which also has a considerably higher than average percentage correct. The Great Basin study area is markedly more topographically complex than the others, with more than twice as many polygons and the most unique classes generated by the automated classification method. This resulted in the lowest percentage correct out of the five study areas.

**Discussion**

The data from the verification process seem to show that the model yields the most accurate results in areas of either high or low relief and is most often confused in transitional topographies. The results also indicate that the model yields the most accurate results from input DEM data that have less topographic variation. These results make sense as observations about the automated model, but in this case they also indicate weaknesses in the verification process. The heads up mapping method used in this study produced an average of 11% of the amount of polygons produced by the automated model, indicating that the scale of the heads-up mapping process was significantly smaller and more generalized than the operating scale of the model. The fact that percentages correct were lower for more topographically complex study areas and higher for classes with more extreme relief could be a symptom of the verification method and not of a weakness in the model itself, as simpler topography or more obvious relief leads to more correct decisions on the part of a human mapper. A dataset of classified landforms agreed upon by the discipline of geomorphology would be the ideal verification dataset for this model, but such a dataset does not exist.

Despite the limitations inherent in the verification method, the results of this study are encouraging for the ongoing use and improvement of this model. An overall rate of agreement of 39% is only 4 points off from the only other comparable study, in which Klingseisen et al. (2007) achieved an initial finding of a 43% rate of agreement between their automated landform classification utility and a heads-up expert classification using an error matrix to test for confusion against a heads-up expert classification. The study area used by Klingseisen et al. (2007) to test their landform mapping software was a farm in Western Australia, considerably less topographically diverse than the five study areas used in this investigation, which might have contributed to their more favorable initial result.

**Further Work**

While LFMapper is proven effective, it remains far from incorporating many of the cutting edge developments in landform classification as they are outside of the scope of this study or of the powers of the ArcGIS software framework. A parallel approach that could be accomplished within ArcGIS is pattern recognition classification (Jasiewicz & Stepinski 2012), which uses a computer visualization method to classify imagery as opposed to the geometrical approach used in this study. Due to the parallel but opposed nature of this approach, it would also be instructive to verify these two methods against each other using an error matrix procedure like the one used in this study. A supervised classification approach, which uses a similar geometric classification method with added step of using geomorphological knowledge to “train” the classification scheme to handle the topographies of specific regions (Veronesi & Hurni 2014), could also be added to an automated process, but the regional specificity would necessitate the need for multiple versions of a custom tool.

The advanced mathematics of fuzzy sets and systems have also been applied significantly to the problem of mapping and classifying landforms and might reflect the geomorphological reality best due to the “fuzzy” boundaries they allow for, however, at this time these approaches require additional statistical software to achieve their classifications (Burrough, Van Gaans & MacMillan 2000), (MacMillan, Pettapiece, Nolan, & Goddard 2000), (Schmidt & Hewitt 2004).

In addition to developing automated versions of these other methods within the framework of ArcGIS, it would always be ideal to further verify LFMapper against other current automated landform classification utilities to see where improvements can continue to be made.

**Conclusions**

The LFMapper custom tool for Esri’s ArcGIS software, based on an improved version of Morgan and Lesh’s 2005 geoprocessing model for the classification of landforms, represents something that did not exist before: an easily distributable, research based, mathematically rigorous tool for use within ArcGIS that allows for the classification of landforms with minimal input or background knowledge from the user. Proper verification of the model presented a challenge as an agreed upon dataset for classified landforms does not exist, however through a verification process used in similar studies, a comparable result was achieved (Klingseisen et al. 2007). This tool will be useful for those interested in using ArcGIS to study landforms and their interaction with other spatial phenomena.

**Works Cited**

Burrough, P.A, P.F.M Van Gaans, R.A. MacMillan. "High-Resolution Landform Classification Using Fuzzy k-Means" Fuzzy Sets and Systems 113 (2000) 37-52

Dikau, Richard, Earl E. Brabb, Robert M. Mark. "Landform Classification of New Mexico By Computer" Open-File Report Number 91-634 (1990) USDI USGS.

Dikau, Richard. “The Application of a Digital Relief Model to Landform Analysis in Geomorphology” Three Dimensional Applications in Geographical Information Systems (1989) 51-77

Fenneman, Nevin M. “Physiographic Boundaries Within the United States” Annals of The Association of American Geographers, Vol. 4, (1914) pp. 84 – 134

Hammond, Edwin H. “Analysis of Properties in Land Form Geography: An Application to Broad-Scale Land Form Mapping” Annals of the Association of American Geographers, Vol. 54, No. 1 (1964) pp. 11 – 19

Hammond, Edwin H. “Small-Scale Continental Landform Maps” Annals of the Association of American Geographers, Vol. 44, No. 1 (Mar. 1954) pp. 33 – 42

Jasiewicz, Jaroslaw, Tomasz F. Stepinski. “Geomorphons – A Pattern Recognition Approach to Classification and Mapping of Landforms” Geomorphology 182 (2013) 147-156

Klingseisen, Bernhard, Graciela Metternicht, Gernot Paulus. "Geomorphometric Landscape Analysis Using a Semi-Automated GIS Approach" Environmental Modeling & Software 20 (2007) 1-13

MacMillan, R.A., R. Keith Jones, David H. McNabb. "Defining a Heirarchy of Spatial Entities for Environmental Analysis and Modelling using Digital Elevation Models (DEMS)" Computers, Environment and Urban Systems 28 (2004) 175-200

MacMillan, R.A., W.W. Pettapiece, S.C. Nolan, T.W. Goddard. “A Generic Procedure for Automatically Segmenting Landforms into Landform Elements Using DEMs, Heuristic Rules and Fuzzy Logic” Fuzzy Sets and Systems 113 (2000) 81 – 109

Minar, Jozef, Ian S. Evans. "Elementary Forms for Land Surface Segmentation: The Theoretical Basis of Terrain Analysis and Geomorphological Mapping" Geomorphology 95 (2008) 236-259

Morgan, John M., Ashley M. Lesh. "Developing Landform Maps Using ESRI's ModelBuilder". (paper presented at the ESRI user conference, 2005)

Pike, Richard J. “The Geometric Signature: Quantifying Landslide Terrain Types from Digital Elevation Models” Mathematical Geology 20 (1988) (5) 491-511

Savigear, R. A. G. “A Technique of Morphological Mapping” Annals of the Association of American Geographers, Vol. 55, No. 3 (Sept. 1965) pp. 514 - 538

Schmidt, Jochen, Allan Hewitt. “Fuzzy Land Element Classification from DTMs Based on Geometry and Terrain Position” Geoderma 121 (2004) pp. 234 - 256

Veronesi, F., L. Hurni. “Random Forest with Semantic Tie Points for Classifiying Landforms and Creating Rigorous Shaded Relief Representations” Geomorphology 224 (2014) 152-160

Zawawi, Azita Ahmad, Masami Shiba, Noor Janatun Naim Jemali. “Landform Classification for Site Evaluation and Forest Planning: Integration between Scientific Approach and Traditional Concept” Sains Malaysiana 43 (3) (2014): 349 – 358

**Refining an Automated Model for Basic Landform Classification**

**A Python and ArcGIS Approach**

Josh Moss

**Appendix A: LFMapper Source Code**

# this script powers a tool that accepts a DEM file as the input and returns a vector dataset of

# landforms classified by the system developed by the Missouri Resource Assessment Partnership (MORAP)

# this script uses "Developing Landform Maps Using ESRI's Modelbuilder" by Dr. John Morgan and Ashley Lesh (2005)

# as a reference

import arcpy

from arcpy.sa import \*

arcpy.AddMessage("Imported Spatial Analyst extension")

# define workspace, operation DEM, and other variable names (remap values, neighborhood settings)

try:

arcpy.env.workspace = arcpy.GetParameterAsText(0)

opDEM = arcpy.GetParameterAsText(1)

landFormPoly = arcpy.GetParameterAsText(2)

hillShadeLayer = arcpy.GetParameterAsText(3)

canyonOption = arcpy.GetParameterAsText(4)

canyonLayer = arcpy.GetParameterAsText(5)

remapOne = RemapRange([[0, 9999, 1]])

remapTwo = RemapRange([[0, 7, 1], [7, 90, 0]])

remapThree = RemapRange([[0.5, 1.0, 10], [0.0, 0.5, 20]])

remapFour = RemapRange([[0, 15, 1], [15, 30, 2], [30, 90, 3], [90, 150, 4], [150, 300, 5], [300, 900, 6], [900, 9999, 7]])

remapFive = RemapRange([[0, 800, 1], [800, 950, 2], [950, 1150, 3], [1150, 1300, 4], [1300, 1550, 5], [1550, 10000, 6]])

remapSix = RemapValue([[1, 3], [2, 3], [3, 1], [4,1], [5,1], [6,1]])

nbrValsOne = NbrCircle(50, "CELL")

nbrValsTwo = NbrCircle(50, "CELL")

print "Variables Defined"

arcpy.AddMessage("Variables Defined")

# Check out the Spatial Analyst extension

arcpy.CheckOutExtension("Spatial")

print "Loaded extension"

arcpy.AddMessage("Loaded extension")

# add 100 to opDEM to avoid issues with negative integers

mapOneHundo = Plus(opDEM, 100)

mapOneHundo.save()

print "Adjusted for negative integers"

arcpy.AddMessage("Adjusted for negative integers")

# reclassify operation DEM so that all cells equal 1 (mapOne)

mapOne = Reclassify(mapOneHundo, "Value", remapOne)

mapOne.save()

print "Reclassified DEM for 'all cells equal 1' raster"

arcpy.AddMessage("Reclassified DEM for 'all cells equal 1' raster")

# run focal statistics for sum on map 1 with a 50 pixel radius circular window (mapTwo)

mapTwo = FocalStatistics(mapOne, nbrValsOne, "SUM")

mapTwo.save()

print "Ran 50 pixel circular window focal statistics"

arcpy.AddMessage("Ran 50 pixel circular window focal statistics")

# run map algebra float operation on map 2 (mapThree)

mapThree = Float(mapTwo)

mapThree.save()

print "Ran float operation"

arcpy.AddMessage("Ran float operation")

# run SA slope tool on operation DEM (mapFour)

mapFour = Slope(opDEM)

mapFour.save()

print "Created slope map"

arcpy.AddMessage("Created slope map")

# reclassify slope map [map 4] resulting in slope categories map (mapFive)

mapFive = Reclassify(mapFour, "Value", remapTwo)

mapFive.save()

print "Reclassified slope map"

arcpy.AddMessage("Reclassified slope map")

# run focal statistics tool for sum on map 5 with a 1.5 KM circular window (mapSix)

mapSix = FocalStatistics(mapFive, nbrValsTwo, "SUM")

mapSix.save()

print "Ran 1.5 KM circular window focal statisitics"

arcpy.AddMessage("Ran 1.5 KM circular window focal statisitics")

# run map algebra to divide map 6 by map 3 (mapSeven)

mapSeven = Divide(mapSix, mapThree)

mapSeven.save()

print "Ran divide operation"

arcpy.AddMessage("Ran divide operation")

# reclassify map 7 with the following remap table: 10 = 0.5-1.0, 20 = 0.0-0.5 (mapEight)[slope parameter]

mapEight = Reclassify(mapSeven, "Value", remapThree)

mapEight.save()

print "Created slope parameter class"

arcpy.AddMessage("Created slope parameter class")

# run focal statistics for maximum value on operation DEM within a 20 pixel radius circular window (mapNine)

mapNine = FocalStatistics(opDEM, nbrValsOne, "MAXIMUM")

mapNine.save()

print "Ran relief parameter focal statistics step one"

arcpy.AddMessage("Ran relief parameter focal statistics step one")

# run focal statistics for minimum value on operation DEM within a 20 pixel radius circular window (mapTen)

mapTen = FocalStatistics(opDEM, nbrValsOne, "MINIMUM")

mapTen.save()

print "Ran relief parameter focal statistics step two"

arcpy.AddMessage("Ran relief parameter focal statistics step two")

# run map algebra to subract map 10 (minimun op DEM value) from map 9 (maximum op DEM value)(mapEleven)

mapEleven = Minus(mapNine, mapTen)

mapEleven.save()

print "Ran relief parameter subtract"

arcpy.AddMessage("Ran relief parameter subtract")

# reclassify map 11 with the following remap table, resulting in (mapTwelve)[relief parameter]:

#1 = 0-15 m

#2 = 15-30 m

#3 = 30-90 m

#4 = 90-150 m

#5 = 150-300 m

#6 = 300-900 m

#7 = 900-max m

mapTwelve = Reclassify(mapEleven, "Value", remapFour)

mapTwelve.save()

print "Created relief parameter"

arcpy.AddMessage("Created relief parameter")

# arrive at classifications by running map algebra to add mapEight (slope parameter) to mapTwelve (relief parameter) for (mapThirteen)[landform map]

mapThirteen = Plus(mapEight, mapTwelve)

mapThirteen.save()

print "Add relief parameter to slope parameter"

arcpy.AddMessage("Add relief parameter to slope parameter")

# smooth appearance of landform map using majority filter tool with 8 neighbors (mapFourteen)

mapFourteen = MajorityFilter(mapThirteen, "EIGHT")

mapFourteen.save()

print "Ran majority filter"

arcpy.AddMessage("Ran majority filter")

# use map algebra to multiply mapFourteen by map 1 (all cells equal 1 map) for a binary masking effect, correcting

# for edge effects from study area size/focal stats operations (mapFifteen)

mapFifteen = Times(mapOne, mapFourteen)

mapFifteen.save()

print "Multiplied by 'all cells equal 1' raster/correct edge effects"

arcpy.AddMessage("Multiplied by 'all cells equal 1' raster/correct edge effects")

# if applicable, activate canyon option reclassification procedure

if str(canyonOption) == "true":

canyonsOne = Reclassify(opDEM, "Value", remapFive)

canyonsOne.save()

print "Created relative elevation zones"

arcpy.AddMessage("Created relative elevation zones")

canyonsTwo = Reclassify(canyonsOne, "Value", remapSix)

canyonsTwo.save()

print "Created low elevation mask"

arcpy.AddMessage("Created low elevation mask")

canyonsThree = Times(canyonsTwo, mapFifteen)

canyonsThree.save()

print "Created canyons raster"

arcpy.AddMessage("Created canyons raster")

arcpy.RasterToPolygon\_conversion(canyonsThree, canyonLayer)

print "Created canyons layer"

arcpy.AddMessage("Created canyons layer")

#add landform names field to attribute table landform polygon feature class (will not be hard coded in final script)

arcpy.AddField\_management(canyonLayer, "landform\_name", "TEXT", field\_length = 50)

print "added field"

arcpy.AddMessage("Added field")

nameField = "landform\_name"

gridCode = "gridcode"

print "defined update fields"

arcpy.AddMessage("Defined update fields")

#create update cursor

with arcpy.da.UpdateCursor(canyonLayer, (gridCode, nameField)) as updateRows:

for lForms in updateRows:

gridCodes = lForms[0]

print gridCodes

print "printed gridcodes"

def codeTextAssign(gridCodes):

if gridCodes == 11:

text = "Flat Plains"

elif gridCodes == 33:

text = "Flat Plains"

elif gridCodes == 12:

text = "Smooth Plains"

elif gridCodes == 36:

text = "Smooth Plains"

elif gridCodes == 13:

text = "Irregular Plains"

elif gridCodes == 39:

text = "Irregular Plains"

elif gridCodes == 14:

text = "Plains with Low Hills"

elif gridCodes == 42:

text = "Plains with Low Hills"

elif gridCodes == 15:

text = "Plains with Hills"

elif gridCodes == 45:

text = "Plains with Hills"

elif gridCodes == 16:

text = "Plains with Low Mountains"

elif gridCodes == 48:

text = "Plains with Low Mountains"

elif gridCodes == 17:

text = "Plains with Mountains"

elif gridCodes == 51:

text = "Plains with Mountains"

elif gridCodes == 21:

text = "Rough Plains"

elif gridCodes == 63:

text = "Rough Plains"

elif gridCodes == 22:

text = "Rugged Plains"

elif gridCodes == 66:

text = "Rugged Plains"

elif gridCodes == 23:

text = "Breaks"

elif gridCodes == 69:

text = "Breaks"

elif gridCodes == 24:

text = "Low Hills"

elif gridCodes == 72:

text = "Low Hills"

elif gridCodes == 25:

text = "Hills"

elif gridCodes == 26:

text = "Low Mountains"

elif gridCodes == 27:

text = "Mountains"

elif gridCodes == 75:

text = "Arroyo"

elif gridCodes == 78:

text = "Canyon"

elif gridCodes == 81:

text = "Steep Canyon"

return text

# define "text" as relationship for updating rows

textLandForm = codeTextAssign(gridCodes)

print str(textLandForm)

print 'printed landforms'

#update rows

lForms[1] = str(textLandForm)

updateRows.updateRow(lForms)

print "Added landform names to attribute table"

arcpy.AddMessage("Added landform names to attribute table")

# run SA hillshade tool on operation DEM for a comparative layer overwhich the MORAP landform polygons can be viewed

hillShade = Hillshade(opDEM)

hillShade.save(hillShadeLayer)

arcpy.AddMessage("Created hillshade layer")

print "Created hillshade layer"

# check in the Spatial Analyst extension after operation is complete

arcpy.CheckInExtension("Spatial")

# add geoprocessing message saying operation was a success

print "Land Form Classification (Canyon Option) operation was successful"

arcpy.AddMessage("Land Form Classification (Canyon Option) operation was successful")

else:

# use raster to polygon tool to convert map 15 into polygon layer

arcpy.RasterToPolygon\_conversion(mapFifteen, landFormPoly)

print "created landforms polygon layer"

arcpy.AddMessage("Created landforms polygon layer")

# add field and update field with text landform names

#add landform names field to attribute table landform polygon feature class

fc = landFormPoly

arcpy.AddField\_management(fc, "landform\_name", "TEXT", field\_length = 50)

print "added field"

arcpy.AddMessage("Added field")

nameField = "landform\_name"

gridCode = "gridcode"

print "defined update fields"

#create update cursor

with arcpy.da.UpdateCursor(fc, (gridCode, nameField)) as updateRows:

for lForms in updateRows:

gridCodes = lForms[0]

print gridCodes

print "printed gridcodes"

def codeTextAssign(gridCodes):

if gridCodes == 11:

text = "Flat Plains"

elif gridCodes == 12:

text = "Smooth Plains"

elif gridCodes == 13:

text = "Irregular Plains"

elif gridCodes == 14:

text = "Plains with Low Hills"

elif gridCodes == 15:

text = "Plains with Hills"

elif gridCodes == 16:

text = "Plains with Low Mountains"

elif gridCodes == 17:

text = "Plains with Mountains"

elif gridCodes == 21:

text = "Rough Plains"

elif gridCodes == 22:

text = "Rugged Plains"

elif gridCodes == 23:

text = "Breaks"

elif gridCodes == 24:

text = "Low Hills"

elif gridCodes == 25:

text = "Hills"

elif gridCodes == 26:

text = "Low Mountains"

elif gridCodes == 27:

text = "Mountains"

return text

# define "text" as relationship for updating rows

textLandForm = codeTextAssign(gridCodes)

print str(textLandForm)

print "printed landforms"

#update rows

lForms[1] = str(textLandForm)

updateRows.updateRow(lForms)

print "Added landform names to attribute table"

arcpy.AddMessage("Added landform names to attribute table")

# run SA hillshade tool on operation DEM for a comparative layer overwhich the MORAP landform polygons can be viewed

hillShade = Hillshade(opDEM)

hillShade.save(hillShadeLayer)

print "Created hillshade layer"

arcpy.AddMessage("Created hillshade layer")

# check in the Spatial Analyst extension after operation is complete

arcpy.CheckInExtension("Spatial")

# add geoprocessing message saying operation was a success

print "Land Form Classification operation was successful"

arcpy.AddMessage("Land Form Classification operation was successful")

except:

# custom error

arcpy.AddError("Land Form Classification operation failed")

# ESRI errors

arcpy.AddMessage(arcpy.GetMessages())