

# **Priority Assessment and Ranking of Conservation-Eligible Lands (PARCEL):**

## **A Flexible GIS Tool for Watershed-Scale Land Trusts**



by

Eugene Boris Yacobson

Dr. Mukesh Kumar, Advisor

May 2013

Master's Project submitted in partial fulfillment of the  
requirements for the Master of Environmental Management degree in  
the Nicholas School of the Environment at

Duke University

## **Abstract**

The use of geospatial information systems (GIS) by land trusts to locate and prioritize lands critical for conservation has become a crucial step in making the process of land acquisition more strategic, proactive, and fruitful. This project introduces an automated GIS tool scripted in Python for the ESRI ArcGIS environment which allows organizations to readily evaluate a select number of aspects of the conservation value of all property parcels in their target area with a minimum of GIS software literacy.

The tool computes a metric of the extent to which each parcel contributes to connectivity of natural green space, draws corridors between natural patches that pass through the most optimal land uses, models areas underserved by open space, and estimates regions of relatively high sediment erosion and deposition. The analysis, with additional criteria, is demonstrated on the Ellerbe Creek watershed in the Durham, North Carolina area, highlighting the significant conservation potential in this highly impaired urban area.

## **Acknowledgments**

I would like to thank my advisor, Dr. Mukesh Kumar, for his invaluable assistance with and generous devotion of time to the development of this project. I would also like to thank Chris Dreps and the staff and volunteers of the Ellerbe Creek Watershed Association, for providing me with a productive, professionally valuable, and highly enjoyable internship experience which led to the development of the idea for this project. In conjunction with this, I am grateful to Mr. Fred Stanback of the Stanback Internship Program for funding that opportunity. I also owe thanks to the numerous officials at the City of Durham who generously helped me track down and make sense of some of the more elusive GIS data. Finally, I am extremely grateful to my family and my fiancée Megan Thompson for graciously enduring several months' worth of unsolicited mini-lectures about sediment transport capacity.

# Table of Contents

Abstract .....	2
Acknowledgements .....	3
List of Figures/Tables .....	6
Introduction .....	7
a. Applications of greenprinting in the Upper Neuse River basin .....	8
1. Upper Neuse Clean Water Initiative Conservation Plan .....	9
2. Ellerbe Creek Critical Area Protection Plan .....	10
3. Ellerbe Creek Watershed Management Improvement Plan BMPs .....	10
4. Limitations of existing plans .....	11
Objective .....	12
Methods .....	13
a. Study area .....	13
b. Analysis criteria and data sources .....	16
c. Tool functionality and workflows .....	19
1. Modeling of erosion and deposition .....	19
2. Euclidean-distance and least-cost connectivity .....	25
3. Connectivity corridors .....	29
4. Underserved areas .....	31
d. Additional criteria .....	33
Results and Discussion .....	34
a. USPED erosion/deposition .....	34
b. Connectivity and corridors .....	38

c. Alternative prioritizations and comparison to existing plans .....	41
d. Tool limitations and future work .....	43
Conclusion .....	46
References .....	47
Appendix: Python Codes .....	50

## List of Figures/Tables

Figure 1: Land cover map of Ellerbe Creek watershed .....	14
Figure 2: Proportions of land cover types in Ellerbe Creek watershed .....	15
Table 1: Alternative weighting schemes for analysis outcome comparison .....	18
Figure 3: Soil erodibility factor .....	23
Figure 4: Cover management factor .....	24
Figure 5: Distribution of USPED erosion/deposition index values .....	35
Figure 6: Topographic pattern of erosion and deposition .....	36
Figure 7: Topographic pattern of erosion .....	37
Figure 8: General pattern of erosion values .....	38
Figure 9: Mean predicted USPED values by subwatershed .....	39
Figure 10: Habitat patch least-cost connectivity scores in Ellerbe Creek .....	40
Figure 11: Least-cost corridors between 10 largest forest patches .....	41
Table 2: Percentages of parcels in Ellerbe Creek meeting analysis criteria .....	42
Figure 12: Percentages of top parcels meeting criteria under 5 weighting schemes .....	43
Figure 13: Comparison of top identified parcels to existing analyses .....	44

## I. Introduction

The use of geospatial information systems (GIS) by land trusts to locate and prioritize lands critical for conservation has become a crucial step in making the process of land acquisition more efficient and fruitful. Using an array of desirable criteria, an organization with limited resources can use GIS methods to highlight outstanding properties throughout their target area, freeing efforts at land acquisition to focus on detailed field evaluation of these properties, landowner outreach, and other logistics of the process. Moreover, the tangible outputs of a GIS-based prioritization process – measurable indicators of conservation value, maps of where these indicators are found, and parcel rankings – facilitate the articulation of a clear, coherent, and consistent strategy for the land acquisition choices that an organization makes, both to its stakeholders and the public at large. For these reasons, a GIS used in such a context is often referred to as a “decision support tool,” with the emphasis on *support*, as such an approach does not eliminate the need for complex scientific and pragmatic considerations throughout the process, but rather narrows the field of view and focuses such efforts (Trust for Public Land 2005).

The application of GIS modeling to assist communities, nonprofits, and other public organizations in making informed decisions about land conservation priorities has been colorfully dubbed *greenprinting* (Lachman *et al.* 2007). The strategic benefit of greenprinting has been to turn land conservation from a mostly reactive to a more designed approach. Rather than waiting for a parcel of land to reveal itself either through a high-profile development threat or an altruistic landowner reaching out to a conservation group, greenprinting allows for an *a priori* evaluation of such parcels and identification of currently unprotected areas that offer the highest conservation benefit based on

predetermined criteria. Such an approach is especially useful in enabling the consideration of multiple criteria in potentially non-overlapping categories and coming to a consensus about the relative importance of such categories. For example, while water quality protection and preservation of habitat integrity may overlap to some extent, neither of these is guaranteed to overlap strongly with human considerations such as creating a preserve within easy reach of a school or in an area with little publicly available green space. Deciding on goals and coming to consensus about a weighting scheme is at the heart of the process, and is greatly aided by the use of GIS approaches that can create alternative scenarios.

### **Applications of Greenprinting in the Upper Neuse River Basin**

North Carolina's Upper Neuse River basin covers 770 square miles, six counties, and eight municipalities, and includes nine public drinking water reservoirs – Falls Lake, Lake Michie, the Little River reservoir, Lake Holt, Lake Orange, New Hillsborough Lake, Corporation Lake, Lake Ben Johnson, and Lake Rogers – serving more than 600,000 people. Due to long-standing water quality issues created by accelerated development, land conservation as a strategy for watershed protection has been actively undertaken by more than half a dozen nonprofit land trusts in the basin, including the Ellerbe Creek Watershed Association, the Triangle Land Conservancy, and the Tar River Land Conservancy (UNC Environmental Finance Center, 2012). Municipalities, including the City of Durham, have also conducted sophisticated modeling and GIS analyses to identify critical conservation lands. Three of these plans will be reviewed here, as they provide a context for the analytical approach employed by the GIS tool proposed in this project, and they have been

used in the project to identify points of overlap between parcels captured by the tool and those in the plans.

#### Upper Neuse Clean Water Initiative Conservation Plan

The Upper Neuse Clean Water Initiative (UNCWI) Conservation Plan, created by a partnership of 10 nonprofit organizations and the City of Raleigh, has been an exceptionally influential regional plan, as the conservation of parcels scoring a 3 or above on a 0-to-5 scale in the UNCWI analysis is eligible for financial support from the Clean Water Management Trust Fund, a direct appropriation from the North Carolina General Assembly to fund the enhancement of degraded waters (North Carolina General Statutes, Chapter 113A, Article 18). The scoring system in the UNCWI plan is a description strictly of those attributes of the property that impact water quality, and does not attempt to incorporate other relevant characteristics that may be less directly measurable.

The GIS approach in the UNCWI plan is undertaken on a raster scale, meaning a single 0-to-5 score is generated for each cell (one cell being 400 ft<sup>2</sup>) of the study area. Some of the criteria considered are: whether a cell is within 100 ft. of a stream or water body, whether a wetland is present, the hydraulic conductance of the soil, the erosion potential in the cell, whether the land cover is “non-developed,” the presence of significant natural heritage areas, and adjacency to protected land. A parcel-scale tabulation of all cells with values of 3, 4, and 5, weighted by their acreage on the parcel, produces a parcel-scale score (Trust for Public Land 2006).

### Ellerbe Creek Critical Area Protection Plan (CAPP)

This plan, completed in 2010 for City of Durham, was issued specifically for Ellerbe Creek watershed, a subwatershed of the Upper Neuse River basin. The purpose of this plan was to establish possible starting points for a network of protected sites “that might ultimately weave themselves into the fabric of the city,” rather than creating isolated protected areas throughout the watershed (Brown and Caldwell, 2010). To that end, the GIS model used in the plan focuses not only on water quality criteria, but considers also proximity to schools, parks, and greenways. Unlike UNCWI, scoring is on a parcel basis from the outset, with each parcel assigned a “1” if a criterion is met and a “0” otherwise. No weighting scheme is employed, except for extra points reserved for parcels above 50 acres in size or containing 500 ft. or more stream frontage.

After scoring, a qualitative distinction is made between parcels, creating two categories: “keystone parcels,” those at the high end of the ranking which can clearly serve as the basis for larger protected areas, and “urban gems,” parcels which meet many criteria but are too small or in too developed an area to suffice as keystones.

### Ellerbe Creek Watershed Management Improvement Plan (WMIP) BMPs and Restoration

Unlike the other plans used as comparison points, the WMIP is not a strictly GIS-based plan, as it attempts to locate potential sites for stormwater best management practices (BMPs) and stream restoration projects using a combination of modeling and field assessments. For BMP projects, a GIS analysis is used in a complementary way to identify sites more likely to be suitable for such projects, including commercial or industrial areas rather than residential, publicly owned lands, areas located in a neighborhood association, sites accessible by foot or vehicle, and upland areas not located

in a jurisdictional stream or wetland requiring special permitting. For stream restoration projects, such criteria include being on or near school property, being within an existing public park or open space, or being the first project of its kind within a one-mile radius (Brown and Caldwell 2010).

### Limitations of Existing Plans

In the formulation of alternative or supplementary greenprinting analyses, such as the one made possible by the GIS tool proposed here, it is helpful to examine the limitations of the existing plans. The absence of a dynamic weighting scheme or of any weighting scheme at all from existing plans means that it is not possible to consider alternative weightings for criteria and thereby envision multiple conservation scenarios.

The plans also rely largely on overlap analysis and a binary consideration of whether a particular mapped feature of the landscape is present or not. While this capacity for uncovering overlaps of diverse data is a key benefit of GIS technology, it does not harness the full potential of the software, which is able to use iterative processes and complex map algebra, deployed via the use of the Python scripting language, to uncover landscape characteristics relevant to conservation that are not directly addressed through overlap and proximity analysis. For example, a simple proximity analysis might determine that a parcel is valuable for landscape connectivity by virtue of being adjacent to already protected land or within a fixed distance of such land. By contrast, a geoprocessing model using Python, very difficult or impossible to construct in ArcGIS's Model Builder graphical interface, can isolate patches of natural land, quantify their contribution to landscape connectivity, pinpoint parcels hosting the largest proportion of such patches, and draw "connectivity corridors" between them, all in response to user-customized variables.

## **II. Objective**

The purpose of this project is to create an easy-to-use ArcGIS tool for land trusts to employ in identifying parcels in the landscape that offer the highest conservation benefit. The tool is meant to supplement existing analyses, not necessarily to replace them, as it does not focus on overlap or proximity approaches that are relatively straightforward to implement in ArcGIS, but rather introduces a handful of more complex, customizable models that depend on a Python scripting framework for their ease of use. These models include two different versions of a connectivity analysis, an analysis that uses a quantitative method to locate areas underserved by public green space, and a soil erosion model that is an improvement over more traditional approaches. Lastly, the tool enables the user to enter a set of weights for any criteria that have been used to evaluate property parcels in their target area, easily generating alternative prioritizations.

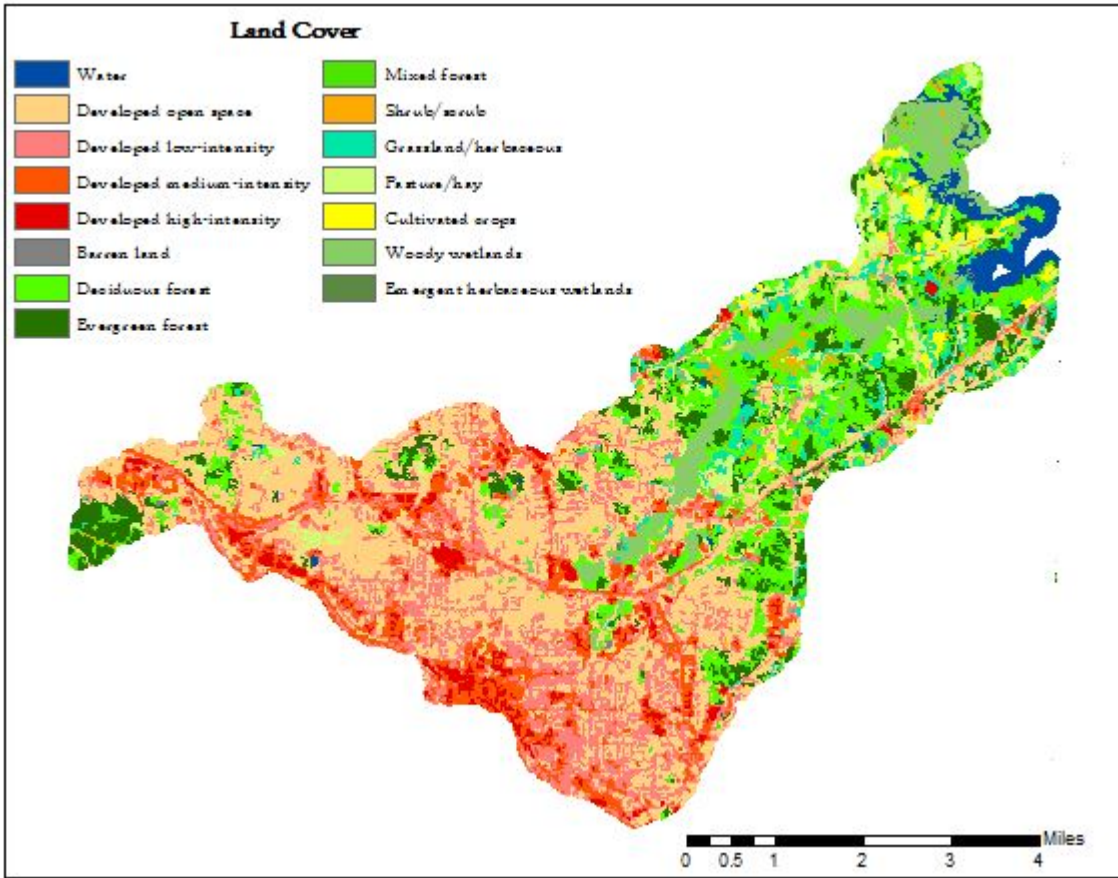
The tool has been used on the Ellerbe Creek watershed to showcase its functionality, using the Python models described above in addition to some simpler analyses performed outside of these models to supplement their outputs. The result is a set of alternative prioritization schemes, based on different weights, of lands in the watershed. These alternative prioritizations also give an idea of the sensitivity of the outputs to different weightings. Lastly, the outputs of the different schemes are analyzed for their degree of overlap with the parcels suggested by the existing plans described above. This comparison provides some insight into the degree of redundancy of the analysis with ones already in use.

### **III. Methods**

#### **Study Area**

The Ellerbe Creek watershed covers an area of 23,526 acres (approximately 37 square miles) of the Upper Neuse River basin, with roughly one-half located within Durham city limits. Its main channel, Ellerbe Creek, flows west to east, much of it following I-85, and enters the Falls Lake reservoir south of the Eno River. Approximately 23.4% of its area is forested, primarily with bottomland hardwood forest and loblolly pine, and 29.8% is developed land where impervious surfaces account for at least 20% of the total cover (Figs. 1 and 2). This watershed is of particular importance to the water quality of the Falls Lake reservoir, as it has the highest percentage of impervious cover of all of the 14-digit watersheds draining into that lake (NCEEP 2006). As has been widely shown in the scientific literature, impervious cover is related to major geomorphic and ecological alterations of the stream system, leading to increased sediment deposition into drinking water reservoirs due to channel erosion by heavy runoff (Leopold *et al.* 2005) and higher downstream nutrient loading that greatly alters community composition (Bernhard and Palmer 2007).

Ellerbe Creek itself has been on North Carolina's 303(d) list as an impaired water body for the list's entire existence (NCEEP 2006). The creek has been shown to transport elevated nutrient loads, total suspended solids, heavy metals, oil and grease, and occasionally fecal coliform bacteria (NCDENR 2001), and for this reason has been declared "biologically impaired" with regard to its designated uses as a water supply water and aquatic life reproduction site. It is estimated that the creek delivers an annual average of 37.8 tons of nitrogen and 4.82 tons of phosphorus to Falls Lake, higher than any other watershed draining to this reservoir (NCEEP 2006).



*Fig. 1. Land use/land cover map of the Ellerbe Creek watershed*

The watershed spans two major geological regions, with its headwaters being in the Carolina Slate Belt, while the rest lies in the Triassic basin. Triassic basin soils are made of deposits eroded from uplands by freshwater, and have low permeability, which promotes low infiltration and high surface runoff during storms. These soils are also highly erodible, making sedimentation a serious concern in this region. In the Ellerbe Creek watershed, modern alluvium deposited by erosion from upstream during the era of heavy agriculture prior to the 1920s is now being actively eroded from streambeds and friable banks as streams incise themselves into the earth. Channelization of Ellerbe Creek by the Army Corps of Engineers has also contributed to sediment loads, by causing bank collapses as the creek attempts to return to its natural meandering morphology (NCEEP 2006).

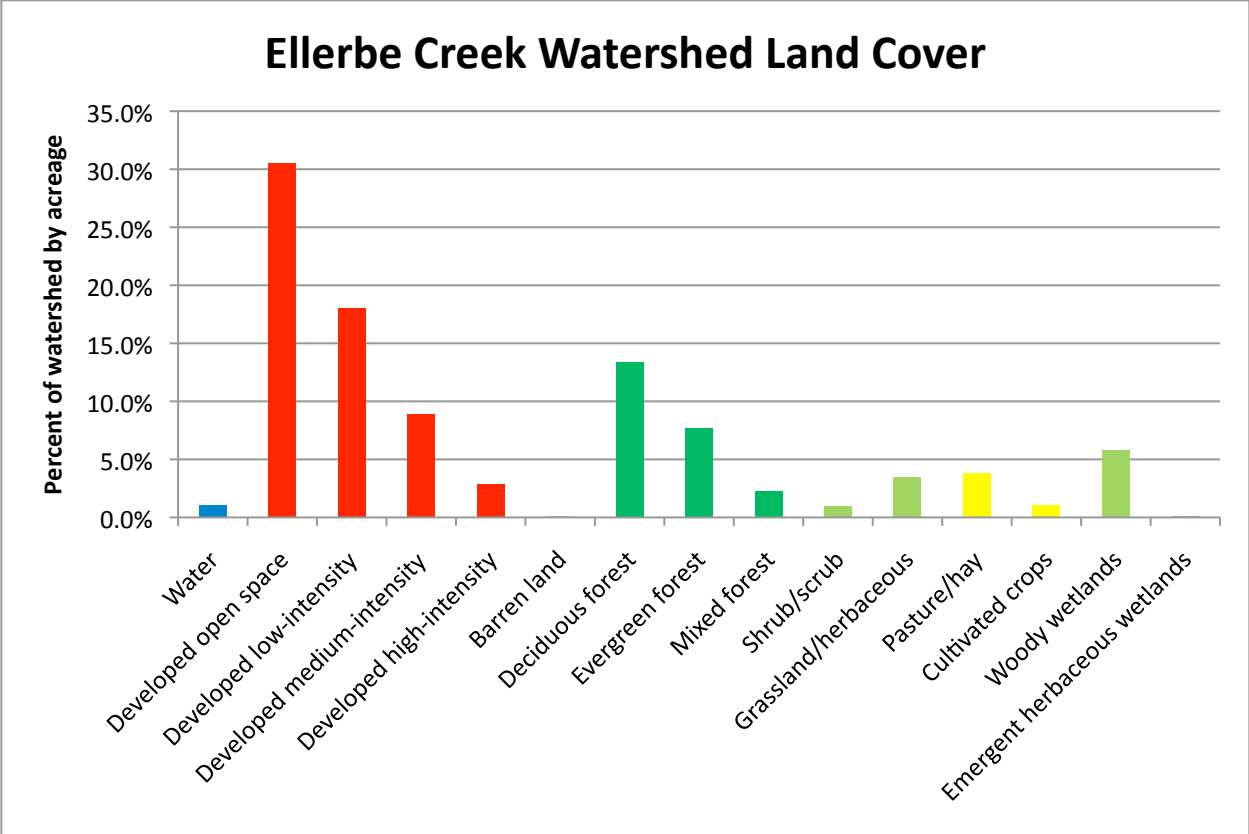


Fig. 2. Proportions of land cover types in Ellerbe Creek watershed.

If protected open space is defined as including public parks, nature preserves (managed by the Ellerbe Creek Watershed Association), federally managed lands, and city-owned vacant lands, there are 283 already protected parcels in the watershed, constituting 3,217 acres (13.7% of watershed area). Excluding city-owned vacant lands, which are not permanently protected from development, this fraction decreases to 2,265 acres (9.6% of watershed area). The total number of parcels in the watershed being 19,985, it seems very probable that untapped conservation opportunities are plentiful in Ellerbe Creek. The Ellerbe Creek Watershed Association (ECWA) currently own and manage 340 acres of preserve land in the watershed, and it is hoped that the GIS tool presented here will help them, and other similar land trusts, periodically refine and reevaluate the conservation opportunities available to them.

## **Analysis Criteria and Data Sources**

Criteria for the analysis presented here are based on ones developed in close collaboration with the Land Acquisition Committee and the Executive Director of the Ellerbe Creek Watershed Association and revised over the course of multiple drafts. Criteria were chosen that were interpretable in spatial terms, mappable with existing GIS data, and likely to select a subset of parcels that was not overly inclusive. The GIS tool developed for this project introduces novel, significantly more complex criteria related to connectivity and sediment erosion modeling than those developed specifically in collaboration with ECWA. The criteria were sorted into three categories relevant to ECWA's conservation mission, presented below (criteria marked with an asterisk are those that have been automated in the GIS tool):

- I. Water quality
  - a. In a high-erosion subcatchment\*
  - b. In a subcatchment with <10% impervious cover
  - c. Within 100 ft. of Ellerbe Creek or major tributaries
- II. Landscape integrity
  - a. Within a user-specified percentile (here, 80<sup>th</sup> percentile) or higher of a metric indicating a parcel's contribution to landscape connectivity, with two alternatives\*:
    - i. A metric that is calculated using Euclidean distances
    - ii. A metric that is calculated using least-cost distances, accounting for passage through optimal types of land cover

- b. In a connectivity corridor connecting user-specified patches (for this example run, the 10 largest contiguous patches of forested land in the watershed were used)
  - c. Contains diabase features
- III. Non-ecological considerations
- a. In an area underserved by publicly available green open space (for this example run, 50<sup>th</sup> percentile or lower of green-space service areas)\*
  - b. Within one quarter-mile of a school
  - c. Within the boundaries of a neighborhood association

Five alternative prioritization schemes were employed to demonstrate potential scenarios generated with the tool when alternative weightings are assigned. The proposed or baseline prioritization is closest to weighting preferences established in consultation with ECWA, in which the water quality category is weighted 45%, landscape integrity is weighted 35%, and non-ecological considerations receive 20%. The other scenarios systematically emphasize one category above the others or set all categories equal. Within each category, the individual criteria are assigned equal weightings (Table 1).

For analyses carried out with the automated tool, an effort was made to use only widely available public datasets, as the tool is meant to be employed flexibly by users across a broad range of potential target areas.

Impervious areas in the Ellerbe Creek watershed (building footprints and paved areas), as well as trails, roads, schools, and neighborhood boundaries were obtained from the Duke University Libraries Durham GIS Features Data Collection (2010). Subcatchments

Table 1. Alternative weighting schemes used to compare possible analysis outcomes.

Criterion	Alternative Weighting Schemes				
	Proposed prioritization (baseline)	Water quality emphasis	Landscape integrity emphasis	Non-ecological considerations emphasis	All equal
In high-erosion subcatchment*	15%	<b>18.75%</b>	13.54%	14.17%	11.11%
In subcatchment with <10% imperviousness	15%	<b>18.75%</b>	13.54%	14.17%	11.11%
Within 100 ft. of Ellerbe Creek or major tributaries	15%	<b>18.75%</b>	13.54%	14.17%	11.11%
In 80th percentile or higher of contributing to landscape connectivity*	11.67%	9.80%	<b>14.59%</b>	10.84%	11.11%
In a connectivity corridor connecting 10 largest patches (excluding patches themselves)*	11.67%	9.80%	<b>14.59%</b>	10.84%	11.11%
Contain diabase features	11.67%	9.80%	<b>14.59%</b>	10.84%	11.11%
In an "underserved area" (50th percentile or lower of green-space service areas)*	6.67%	4.80%	5.21%	<b>8.34%</b>	11.11%
Within 1/4 mile of a school	6.67%	4.80%	5.21%	<b>8.34%</b>	11.11%
Within the boundaries of a neighborhood association	6.67%	4.80%	5.21%	<b>8.34%</b>	11.11%

used as the unit of analysis for finding low-imperviousness and high-erosion subcatchments were those defined in the Ellerbe Creek Watershed Management Improvement Plan (WMIP), with the data obtained from the Durham Department of Public Works, Division of Stormwater Services. The subcatchments were created using detailed field verification methods and serve as the official management units for a number of nonprofit and government entities. Soils data were obtained from the Natural Resource Conservation Service (NRCS) Soil Survey Geographic (SSURGO) database. A 1-arcsecond (30-meter) digital elevation model (DEM) was acquired from the USGS National Elevation Dataset. A land cover raster at 30-meter resolution from 2006 was obtained from the National Land Cover Database developed by the Multi-Resolution Land Characterization Consortium.

Stream hydrography data were provided by the City of Durham GIS Division – this stream dataset has been verified via field methods and aerial photography, and is much

more detailed than the National Hydrography Dataset, although it is discontinuous, due to areas where features have not been fully surveyed and because of its integration with a supplementary shapefile of stormwater pipes and channels. Diabase features were obtained courtesy of the North Carolina Geological Survey.

Protected open space for the definition of underserved areas was assembled from a variety of sources, including shapefiles obtained from the Triangle J Council of Governments (TJCoG) and Ellerbe Creek Watershed Association, as well as derived from parcels data created by the City of Durham GIS Division and shared by TJCoG.

Finally, for comparison with existing analyses, UNCWI Conservation Plan outputs were obtained courtesy of the Triangle Land Conservancy and outputs of the Ellerbe Creek CAPP as well as proposed BMP and stream restoration projects were provided by the Durham Department of Public Works.

## **Tool Functionality and Workflows**

### **Modeling of Erosion and Deposition**

The prediction of soil erosion in GIS is often carried out using empirical models such as the Universal Soil Loss Equation, one of the most common GIS models for calculating estimates of soil erosion (Wischmeier and Smith 1978). The USLE predicts an average rate of sheet and rill erosion for a given kind of soil, management practice, rainfall pattern, and topography. This equation is:

$$SE_i = RK_iLS_iC_iP_i$$

where:

$SE_i$  = the computed soil loss per unit area, expressed in the units resulting from the product  $RK_i$  (the other factors being dimensionless)

$R$  = the rainfall erosivity factor, which is equal to the kinetic energy of a rainfall event times its maximum 30-minute intensity (Brown and Foster 1987).

$K_i$  = the soil erodibility factor in cell  $i$ , a characteristic of specific soil types that is a function of their organic matter content, soil texture, soil structure and permeability; given as a rate of soil loss on a standard plot 22 meters long with a slope of 9%

$C_i$  = cover management factor in cell  $i$ , the ratio of soil loss from land use under the conditions at cell  $i$  to that from continuously fallow and tilled land

$P_i$  = the conservation or support practice factor, being the ratio of soil loss with a specified erosion management practice (e.g. terracing or contouring) to soil loss under “unit-plot” conditions

$LS_i$  = the slope steepness or length factor for cell  $i$ , defined by Moore and Burch (1986) as:

$$LS_i = [A_{Si}/22.13]^n [\sin(\alpha_i)/0.0896]^m$$

where:

$A_{Si}$  = the specific area at cell  $i$ , defined as  $A_{up}/w_n$ , i.e. the upslope contributing area at a cell per unit width normal to the flow direction ( $w_n$ )

$\alpha_i$  = the slope gradient in degrees for cell  $i$

0.0896 = 8.96% = 5.16 degrees, the slope of the standard USLE plot

22.13 = the length of the standard USLE plot in meters

$n$  and  $m$  = empirical coefficients, with typical values around 0.6 and 1.3, respectively.

The USLE is known as a “detachment-limited” method, as it assumes that water can transport an infinite amount of sediment, and that the amount of soil erosion is limited only by the ability of water to detach the soil. Because of this assumption, the USLE does not

predict areas of deposition. According to Mitasova *et al.* (1996), the application of the USLE to landscape-scale erosion modeling is inappropriate, as it was developed for agricultural fields, and it does not consider the influence of terrain shape, rendering it unsuitable for complex topographic conditions. The USLE cannot account for the convergence and divergence of overland flow or for concave and convex slope formations that influence patterns of erosion and deposition.

The Unit Stream Power-Based Erosion/Deposition (USPED) model was developed by Mitasova *et al.* (1996) to address this shortcoming of USLE, and is therefore employed in the proposed tool to simulate the relative magnitudes and spatial patterns of erosion and deposition. This makes possible the identification of management units (subwatersheds or parcels) with the highest erosion potential. In the trial run of the tool presented here, the management units used are subwatersheds, to minimize the effects of spatial error in stream locations and DEM resolution which would likely be prevalent on some of the very small parcel units.

The sediment transport capacity  $T$  in the USPED model is given by the same equation as the sediment erosion  $SE_i$  in the USLE as described above. The slope length factor  $LS_i$  represents the influence of terrain on soil erosion (slope and upslope contributing area) while the other factors account for transportability coefficients dependent on soil and cover. The model then measures the potential on the landscape for erosion and deposition by a topographic index  $E$  that represents the change in sediment transport capacity in the direction of flow – i.e.  $E = dT/ds$ , where  $T$  is the transport capacity.  $dT/ds$  is in turn equal to  $\partial T/\partial x(\cos(\alpha)) + \partial T/\partial y(\sin(\alpha))$ , where  $\alpha$  is the aspect angle computed from the DEM.  $E$  is positive in areas with deposition potential (where  $T$

decreases) and negative in areas with erosion potential (where  $T$  increases) (Mitasova *et al.* 1996). The conceptual idea is that, when transport capacity decreases, water will deposit sediment until its load reaches its new, diminished capacity, and when transport capacity increases, water will erode sediment until its load reaches its new, enhanced capacity.

User inputs into the tool interface include a DEM and the  $R$ ,  $K_i$ ,  $C_i$ , and  $P_i$  factors, as well as a spatially distributed exponent  $n$  if desired, which can reflect the relative differences in the potential for sheet or rill erosion depending on land cover (Leh *et al.* 2011). Denser vegetation cover is likely to inhibit rills, reflected in a lower  $n$  to downplay the influence of the water term  $A_{Si}$ , whereas bare soil is more likely to create flow turbulence and more severe erosion, reflected in a higher  $n$ . The user should also supply the subcatchment or parcel boundaries by which to summarize the model output. The tool then is able to output the mean value of  $E$  per subcatchment or parcel.

For the run described in this report, the value for  $R$  was obtained from a USDA isoerodent map (USDA 1978), given as 250, in units of 100 foot-tons per acre per year. The soil  $K_i$  factor was obtained from an attribute in the SSURGO soil database, given in units of tons of soil per 100 ft-tons of energy (Fig. 3). The product of these yields units for  $T$  of tons per acre per year. The cover management factor is an especially sensitive factor about which there is little scientific consensus, especially in non-agricultural areas such as this study area (Karpilo and Toy, 2003). It is a highly influential factor, as its values can range from nearly 0 to 1 or somewhat higher, exerting a significant effect on the transport capacity estimate (Toy *et al.* 1999). The  $C_i$  factors selected here were based on values derived from the literature and summarized in Leh *et al.* (2011) (Fig. 4). As is often the case, the  $P_i$  factor

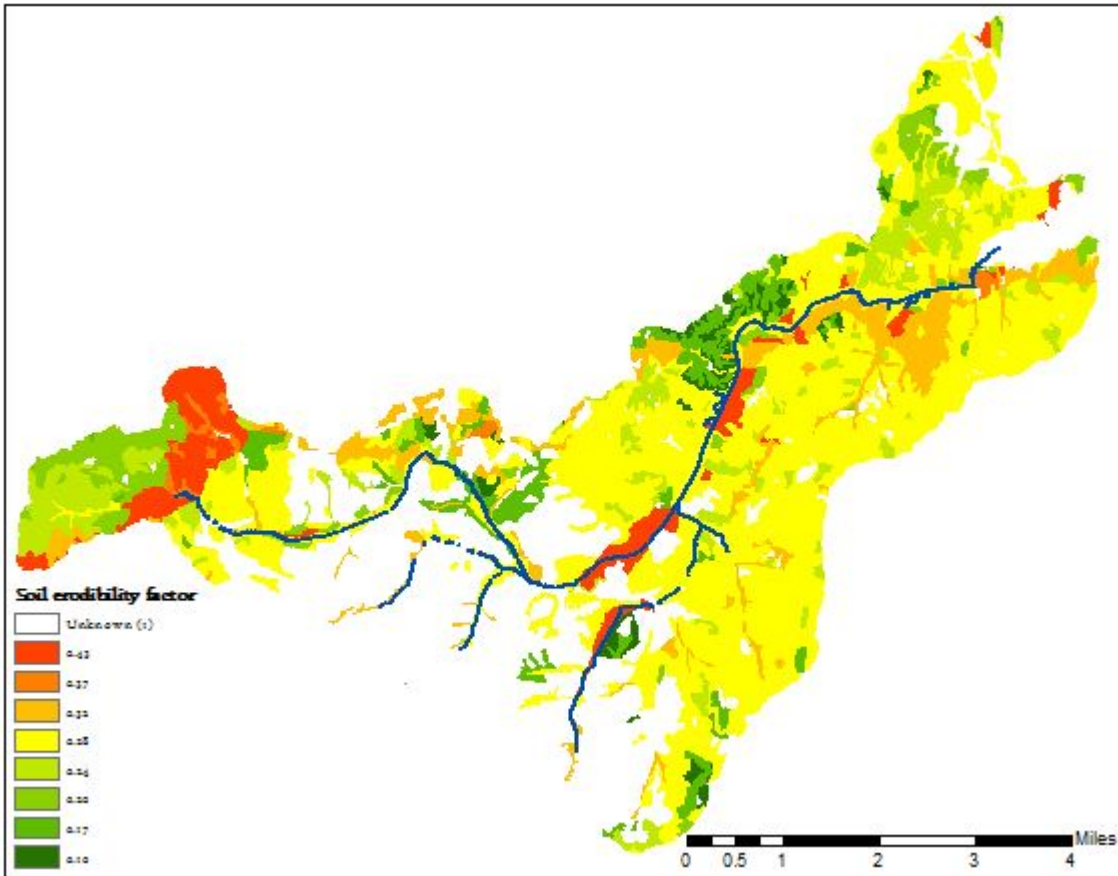
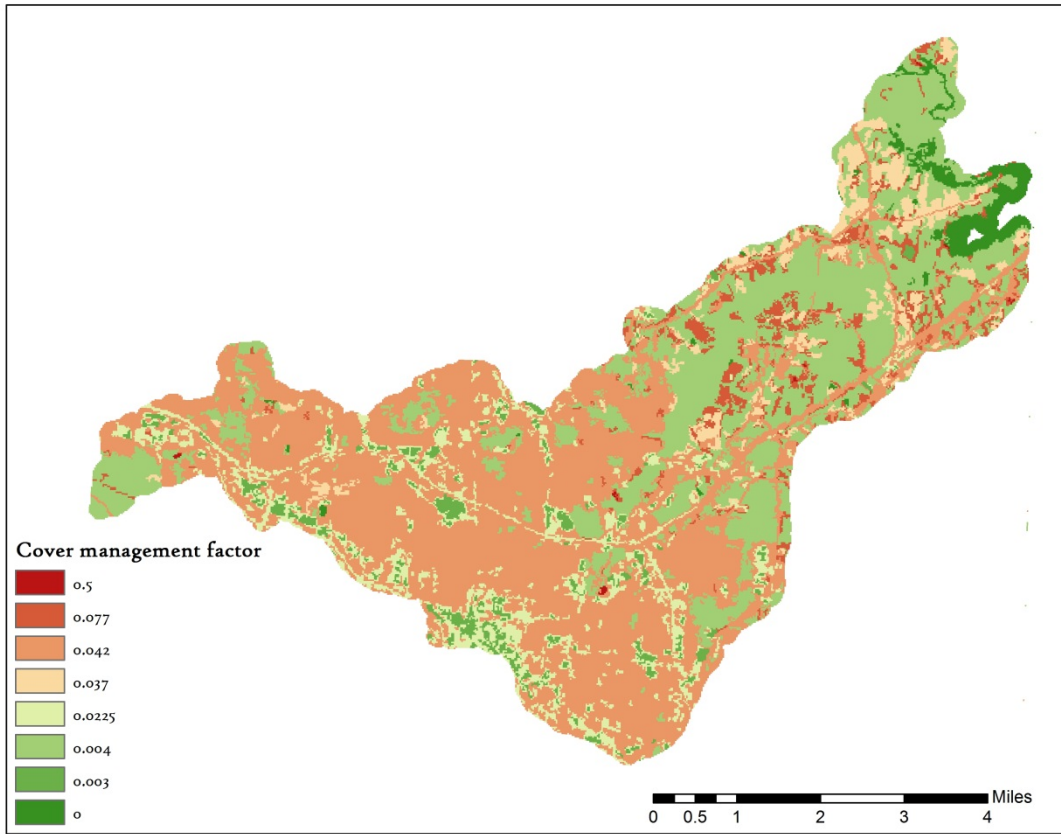


Fig. 3. Soil erodibility factor ( $K$ ). Note  $K$  is unknown for “urban-complex” soils, which have generally not been characterized. It is taken to be the highest  $K$  factor, to reflect exceedingly high sediment production during urban construction activities.

used here was 1, assuming that no special practices (terracing, contouring, hay bales, silt fences, and so on) were being used in a systematic way in this non-agricultural region. A spatially distributed  $n$  and an  $m$  value of 1.3 was used.

$T\cos(\alpha)$  and  $T\sin(\alpha)$  is calculated as specified in the equation for  $E$  above, creating the outputs “qsx” and “qsy.”  $\partial T/\partial x(\cos(\alpha))$  is calculated as  $\cos([\text{aspect of qsx}] * -1 + 450) * \tan([\text{slope of qsx}])$  and  $\partial T/\partial y(\sin(\alpha))$  as

$\sin([\text{aspect of } qsy] * -1 + 450) * \tan([\text{slope of } qsy])$ .  $qsx$  and  $qsy$  represent the divergent sediment transport capacity coefficients in the x and y directions across a cell while  $\partial T / \partial x(\cos(\alpha))$  and  $\partial T / \partial y(\sin(\alpha))$  represent the change in  $T$  in the x and y directions. These



*Fig. 4. Cover management factor (C).*

directional derivatives are then added together, with a more negative result suggesting greater erosion potential and a more positive result suggesting greater deposition potential.

Zonal means per user-supplied summary unit (subwatershed or parcel) are calculated, and a point is awarded to all parcels within a zone where the mean  $E$  is lower than a user-supplied threshold – in the case of this run, -1.0.

### Euclidean-Distance and Least-Cost Connectivity

A commonly stated mission of watershed-scale land trusts, including the Ellerbe Creek Watershed Association, has been the preservation or construction of an interconnected network of green space throughout the target area, one which connects people to nature via recreational infrastructure such as trails and nature to itself by ensuring maximum continuity of green-space patches or proximity to one another. A straightforward and intuitive way to do this in GIS has been simply to locate those parcels which are adjacent to already protected parcels as well as those within some specified straight-line distance, such as one-quarter of a mile.

The limitation to this methodology is that it does not take into account the landscape characteristics of actual patches of forested or other natural land, and in fact does not attempt to locate these patches at all. Parcels adjacent or proximal to protected land may not necessarily be host to natural land of their own. Moreover, not all patches of natural land are equal, as some contribute more to green-space connectivity than others, by virtue of their size, the size of their neighbors, and their proximity to one another. The approach used in this tool attempts to quantify the contribution of each individual parcel to maintaining the integrity of green space.

The measurement of connectivity is a very broad and deep subject and this tool does not attempt to comprehensively evaluate this feature or to replace other tools that more exhaustively characterize the connectivity of a landscape. Nevertheless, it does offer the user a variety of approaches supported by the literature which can be automated in ArcGIS only or mainly through the use of its Python scripting environment. It is also important to keep in mind that, especially in an urbanized context such as the Ellerbe Creek watershed, connectivity is important not only to dispersing animal organisms, but, from a planning

perspective, can enhance the density of human encounters with nature and provide practical possibilities for where to construct trails to enhance the green-space network.

There are a number of isolation and connectivity metrics that are preferred, and no single one has enjoyed widespread acceptance (Moilanen and Hanski 2001). The simplest metrics might use just the distance from a patch to its nearest neighbor (Crooks 2002), while others have found that calculating the average distance to all neighbors, accounting for patch areas, or calculating travel distance through the matrix in such a way that preferences for certain land cover types are accounted for (“least-cost distances”) create more biologically realistic estimates of connectivity (Calabrese and Fagan 2004).

The tool includes both a Euclidean-distance and a least-cost distance approach to measuring connectivity. While Magle *et al.* (2009) have shown that cost-weighted distance metrics perform better than Euclidean-distance metrics in predicting the distribution of dispersing organisms, a Euclidean-distance approach was included due to its significantly faster processing time and the absence of a need to supply a “cost surface” that quantifies the relative desirability of passing through each type of land cover, which may not necessarily be a research effort users will have the resources to undertake.

The area-weighted connectivity metric used in the tool is based on Magle *et al.* (2006) and is represented by the following equation:

$$C = \sum(A_i A_j)^{0.7} / d_{ij}^{1.7}$$

where:

$A_i$  = the area of the “focal” patch (the patch for which connectivity is being calculated)

$A_j$  = the area of another patch in the landscape

$d_{ij}$  = the Euclidean or least-cost distance between the focal patch and another patch  
in the landscape

0.7 and 1.7 are exponents meant to reflect that dispersing organisms respond nonlinearly to changes in isolation/connectivity (Brown *et al.* 2002)

From a user-provided standard MRLC land use/land cover raster, the tool automatically extracts all natural land covers (forested, grassland/herbaceous, scrub/shrub, and wetland) and groups them into discrete features such that all contiguous cells of the selected land cover types form a distinct patch. The areas of each patch are calculated, as well as a table of distances from each patch to every other patch within a user-supplied distance threshold. Joining the area and distance attributes into one table, the metric  $C$  is calculated between each patch and all of the other patches within the dispersal distance. Then, a dissolve operation sums the individual  $C$  values for each patch and all of its neighbors (the  $\sum$  operator in the above equation). Connectivity scores are joined to the patch features and, using a user-supplied parcels shapefile, an identity operation determines, for each parcel, which patches may be found on it, transferring their  $C$  scores to the patch features.

During this transfer, the  $C$  score is reduced in proportion to the fraction of the patch that is actually found on a given parcel. This procedure reflects the fact that a parcel is not “responsible” for hosting all of the  $C$  of a patch found on it when it does not, in fact, contain all of that patch (what could be called the “parcelization” of a patch). Another dissolve operation then sums all the  $C$  scores inherited by each parcel from the patches that are found on them. The outcome of these calculations is generally an extremely small number,

but one which represents, in the context of all the other such values, the relative contribution of each parcel to maintaining the integrity of patches with a certain amount of connectivity.

Finally, the tool asks the user to specify a “percentile” in the form of a value ranging from 0 to 1. This percentile will determine how high a parcel’s *C* score needs to be relative to the others in order to receive a “point,” marking it as a parcel that meets the connectivity criterion. In the trial run presented here, a value of 0.8 was used, meaning that all parcels with a *C* score in the 80<sup>th</sup> percentile or higher were considered to have met the criterion.

Another version of the same script uses least-cost distances rather than straight-line distances. This version of the approach takes into consideration the nature of the matrix between patches. Since the matrix is not homogeneous, a path between two patches that is composed of highly developed commercial/industrial land or a highway can be considered to be “longer” in terms of traversability than a more circuitous path that avoids such undesirable land and instead passes through a forested area. Such a distance may be said to have less of a “cost” than a physically shorter distance through shopping mall parking lots or across a highway. Even outside of the context of the dispersal preferences of a specific species, from the human perspective, a greener matrix between potential preserves could well be more desirable. To calculate such least-cost distances in this tool, the user must supply a cost surface, which is a raster layer where the value of each cell represents the relative cost of traversing the land cover found in that cell. For the purposes of this run, a cost surface was constructed that accounts for land cover as well as assigns significantly higher costs to major roads, with cost decreasing gradually with distance from the road until a distance of 500 meters is reached.

The script iteratively extracts patches one-by-one from the patch raster that it automatically creates and calculates a cost-distance raster for each extracted patch. For each such cost-distance raster, the minimum value in this raster falling within the boundaries of all the other patches is written to a newly created .csv file – this represents the least-cost distance from those patches to the focal patch. Patch areas are calculated and the value of  $C$  is determined as described above, but with this cost-weighted  $d_{ij}$  term instead. While processing time increases significantly due to the iterative nature of these calculations, the amount of information conveyed by the value of  $C$  is now much greater.

It is also possible to significantly adjust the goals of this kind of analysis by altering the cost surface. For instance, if the intention of an organization is to specifically emphasize the value of those patches found in a heavily urbanized matrix, the cost of passing through urban land can be made lower relative to natural land. The higher “connectivity” of urban patches in this case would reflect their status as oases of sorts in an otherwise developed landscape.

### Connectivity Corridors

Due to the fact that urbanization can easily sever the connections between landscape patches, a large body of conservation biology literature focuses on the utility of corridors, which are defined as linear habitat connecting multiple larger habitat blocks that ostensibly enhances the ease of passage between the larger blocks and thereby the viability of populations making use of this habitat (Beier and Noss 1998). The utility of corridors can vary greatly with regard to the specific focal species of interest and their design is a scientifically sophisticated endeavor which can only partially be aided by geospatial analysis methods. The objective of this tool is to provide an automated method to identify

the most traversable land uses between a user-supplied network of habitat patches, again using a cost surface which represents the relative ease of passage through specific types of land cover.

This is a significant refinement over a metric such as buffer proximity to protected land, as it can isolate specific strips of land which, if conserved, would ensure additional landscape integrity in the form of paths between already protected lands – paths which themselves have the maximum habitat value possible given the land cover configuration of the matrix. Again, even in the absence of specific data about the dispersal habits of any one particular species, such corridors could greatly assist in, for instance, ECWA’s mission of creating a watershed-wide green-space network, one that is also accessible to humans. Knowing the location of corridors could, among other things, inform trail network planning or acquisition of key “stepping-stone” preserves between larger preserves.

The tool functions by again creating a patch raster of discrete patches, iteratively extracting them one-by-one, and creating a cost-distance raster for each one. It then cuts a hole in each cost-distance raster in the location of the patch on which it is based, and saves the raster to a unique directory in a user-specified workspace. Then the tool iterates only through the unique pairings of the cost-distance rasters, creating potential corridor rasters between the patches involved in each pair by adding the cost-distance grids together.

The user specifies a cutoff value which represents how many of the unique cost-distance values from a least-to-greatest list of such values should count as part of the corridor, and all cells with a cost-distance value at or below this cutoff are extracted and written to an empty feature class, complete with attributes indicating the “to” and “from” patches of the corridor. In this way, the user is able to control the permissiveness or

restrictedness of the corridors created by the tool. A higher cutoff value creates broader corridors encompassing relatively less suitable land types. Some experimentation may be needed, as a value that is too low may create corridors that fail to “meet in the middle” to connect their origin and destination patches and a value that is too high may create corridors so permissive that their ends resemble buffers. However, realistic corridors may be easily created between a large number of patches with minimal user input, or between specific pairs or subsets of patches in piecemeal fashion, if a universal cutoff value does not provide sensible corridors for all patch pairs.

### Underserved Areas

This portion of the tool attempts to quantify the degree to which areas of the watershed are served by publicly accessible green open space. This is accomplished by subdividing the watershed into Voronoi polygons based on a user-supplied shapefile of open-space areas. Voronoi polygons are formed in such a way that everything contained within a polygon formed around a feature  $x$  (in this case, a park or a preserve) is closer to feature  $x$  than to any other feature of its kind. Voronoi polygons form a tessellation of the watershed area, with each polygon representing a “service area” for the park or preserve that it is based on. The assumption of this method is that people would theoretically prefer to visit parks that are nearest to them and an increase in the availability of such parks would be desirable.

The degree to which a service area is served by the green space it contains is quantified by calculating the amount of green space per capita in each service area, based on a census block shapefile with population attributes provided by the user. A more heavily populated area with less available parkland would be considered underserved. It is

assumed here that a park's role in the landscape is less significant when it is in a more populated area, as on the whole the park is less equipped to serve each individual person.

The main work of this script consists of executing a clustering algorithm, which is meant to prevent a situation in which every individual feature in a parks shapefile is given its own service area. This would likely create an indefensible scenario in which every park polygon, even if it is extremely close to other park polygons, would be assigned its own service area, with the implicit claim that individuals residing in such service areas are bound to their specific green-space polygon even when another one lies, say, 1,000 feet away. To avoid this, the user supplies a "clustering threshold" distance such that two parks within this distance of each other would be grouped into a single feature and thus would receive their own common service area. In many cases, park A will be within the clustering distance of park B, and park C will be within the clustering distance of park B, but not park A. In such a situation, parks A, B, and C will all be assigned to one cluster. In a future revision, it will be possible to filter for "true clusters," in which parks A, B, and C are all within the clustering distance of one another.

This script follows the same general methodology as the others in the toolbox with regard to cutoff values. The user provides a percentile between 0 and 1, and service areas at or below this percentile in their amount of parkland per capita are considered "underserved," with parcels whose centroids are inside these areas receiving a point for being located in an underserved area. In addition, when clipping census blocks to service areas, the population count in the census block is automatically adjusted in proportion to the fraction of the block that is clipped. Thus, if 100 people reside in a census block, and

exactly one half of it is assigned to service area  $x$ , the population estimate for that census block in service area  $x$  would be 50.

### **Additional Criteria**

For the purpose of a more complete prioritization, additional criteria were examined that did not require a complex automated geoprocessing model but could be included through relatively straightforward overlap and proximity operations.

For water quality, the imperviousness percentage of subcatchments was considered. Numerous studies over more than three decades have shown remarkably consistent results relating aquatic system conditions with catchment development level (Booth and Jackson 1997), with the onset of degradation tending to be around 10-15% total impervious area of a stream catchment. Flashy hydrographs, high pollutant loads, and more homogeneous habitat have been strongly connected to species-poor urban fish and invertebrate assemblages (Morgan and Cushman 2005). At the same time, macroinvertebrate diversity appears to become consistently low once impervious surfaces in the watershed exceed 10-15% (Klein 1979). Therefore, parcels in subwatersheds with <10% imperviousness were flagged with a point.

As an initial attempt to identify potential nutrient sinks in the watershed, parcels within 100 feet of the Ellerbe Creek and/or its major tributaries were weighted with a point. In a future revision, improved methods of buffer identification may be made by looking at the full extent of contiguous vegetated buffer around a stream, rather than a fixed-distance buffer, and giving extra weight to those buffers located in especially high-accumulation flowpaths approaching the stream network, according to the strategies described in Baker *et al.* (2006).

In the category of landscape integrity, a feature of potential interest for the preservation of rare and unique natural characteristics and species diversity is the presence of diabase intrusions. Diabase is a type of igneous rock which occurs in sheet formations called dikes and sills, formed by the intrusion of molten magma into existing rock. The weathering of diabase leads to the formation of unusually basic soils (at least within the North Carolina Piedmont), known to be associated with rare, species-rich communities and species, such as the federally endangered smooth coneflower (*Echinacea laevigata*) (Walker 2009). In addition, diabase formations are somewhat relevant to water quality, as diabase-rock derived soils generally have poor infiltration rates and high runoff potential, making preservation of vegetated buffers and lower impervious surface in such regions of greater importance. A GIS coverage of diabase regions at 1:24,000 scale was therefore overlapped with watershed parcels.

Lastly, for non-ecological considerations in addition to parcels in underserved areas, parcels were flagged for being within one-quarter mile of a school or within the boundaries of a neighborhood association, the former providing greater community access opportunities and the latter possibly being easier to acquire through potential partnerships with the association.

## **IV. Results and Discussion**

### **USPED Erosion/Deposition**

According to Mitas and Mitasova (1998), the predominant USPED values in relatively well-preserved areas that are not significant hotspots of erosion or deposition will be found in the interval (-1,1). The histogram of values from the raster representing USPED's output of *E* magnitudes for Ellerbe Creek conforms to this pattern, with the

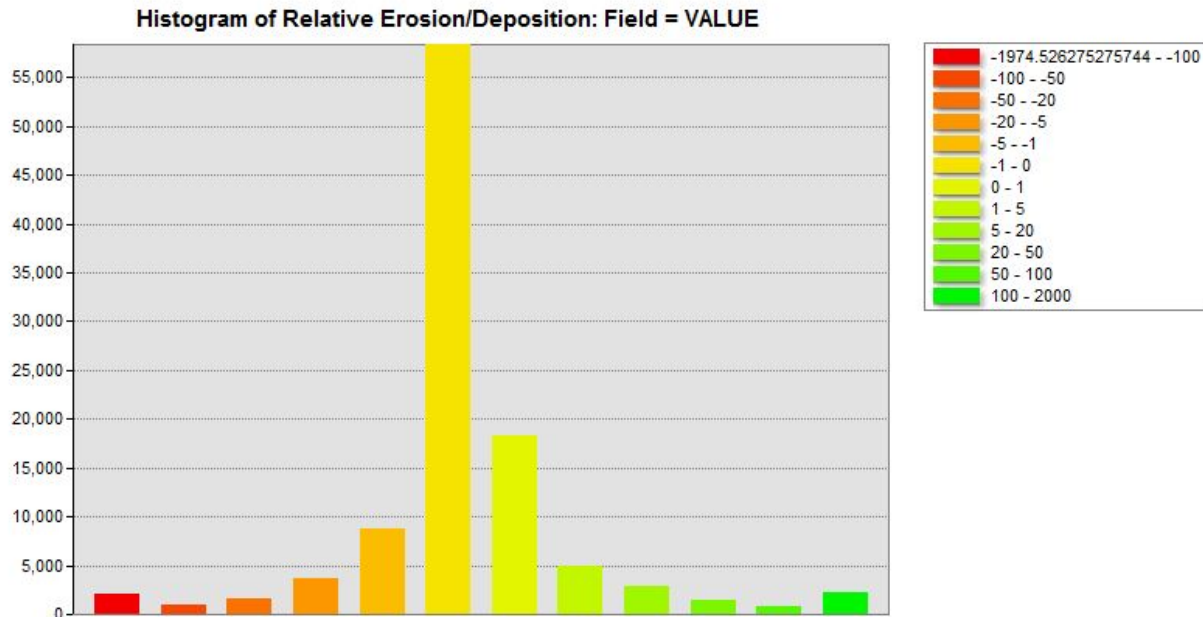
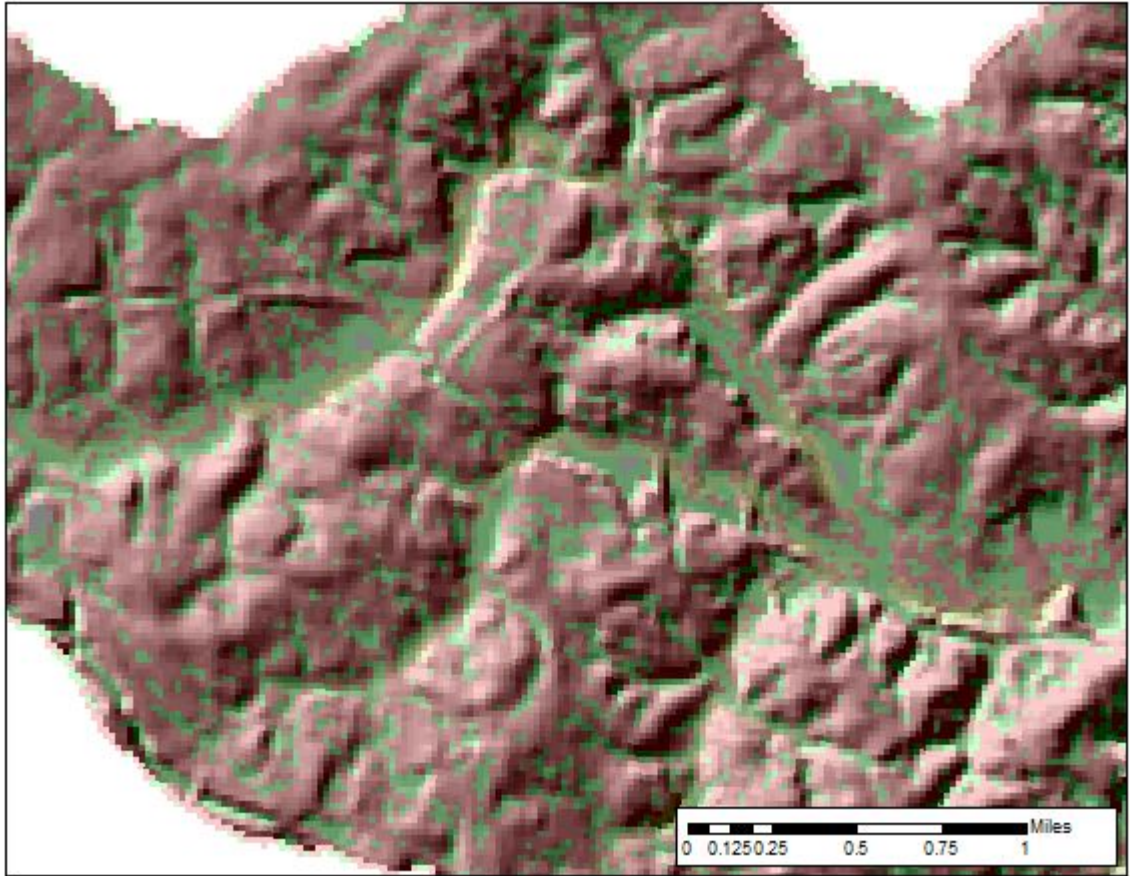


Fig. 5. Distribution of USPED erosion/deposition values.

majority of values found in this  $(-1,1)$  interval (Fig. 5). From this histogram, erosion appears to be predominant in the watershed, with the most populated bin being  $(-1,0)$ . This concentration of values within a very narrow range in the middle, with long tails covering a much greater range, is typical of USPED results found in the literature.

A close-up map of the general pattern of erosion and deposition over a topographic hillshade reveals an intuitive pattern, with erosion occurring on slopes and deposition in valleys (Fig. 6). The fact that erosion occurs specifically on slopes and not on hilltops is significantly more evident in Fig. 7, where the range of values is set to span  $(0,-1)$ , with redder areas closer to  $-1$ . The watershed as a whole Fig. 8 appears to have greater erosion in the urban areas (interval  $(0,-1)$ ), which is perhaps consistent with the higher  $C$  factor in these areas and the high  $K$  assigned to urban-complex soils to reflect the presence of sediment-producing construction sites in the urban environment. Therefore, when  $T$  behaves in an erosive way in this region, the erosive effect is ultimately higher.



*Fig. 6. Pattern of erosion and deposition in western portion of Ellerbe Creek topography. Red reflects erosion from hills and green deposition into depressions.*

Looking at the full range of values, it may be found that the highest erosive regions are found in areas of concentrated flow, where higher-order major tributaries and Ellerbe Creek itself create high transport capacities (Fig. 9). Mitsova and Mitas (1999) express some doubt about the validity of the much higher erosional zones that USPED systematically predicts in areas of concentrated flow – they suggest that, while these predictions may be correct, field verification is strongly recommended. It is also recommended that lower exponents of  $m$  and  $n$  be used, which may be especially applicable in the case of this trial analysis.

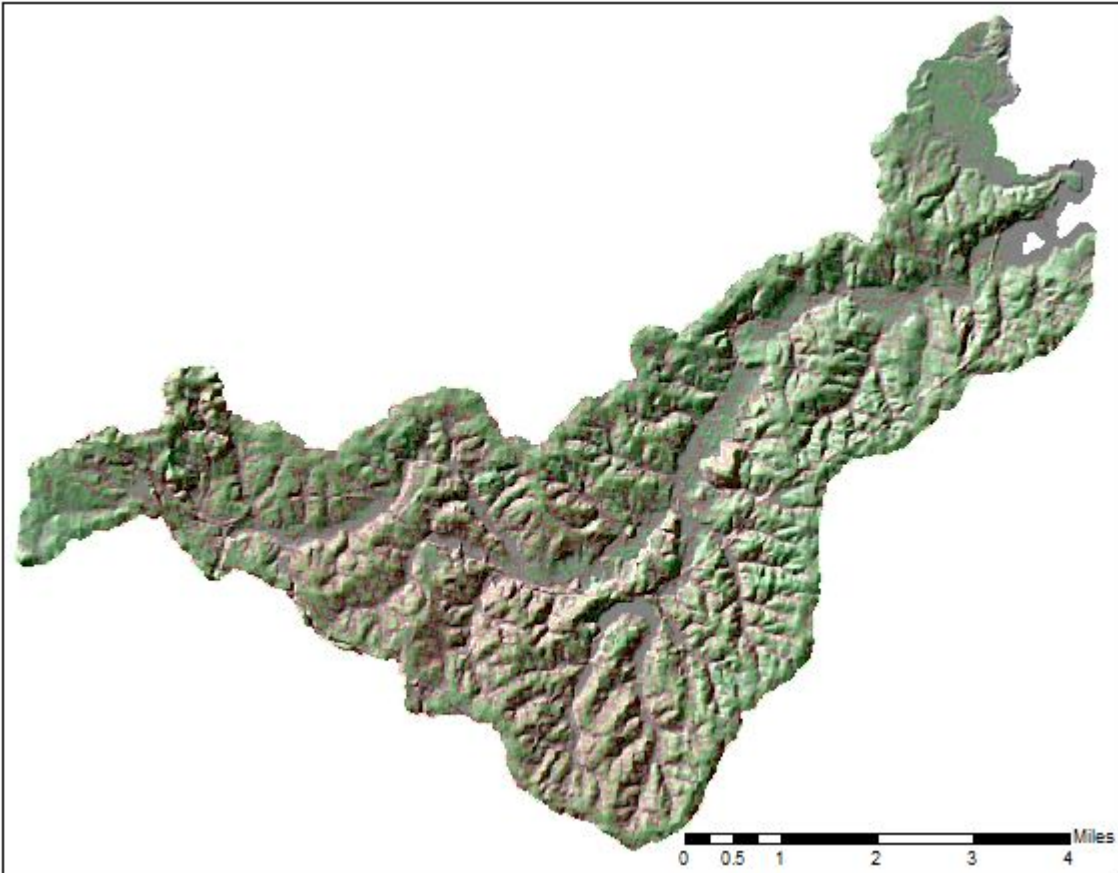


*Fig. 7. Erosion values between 0 and -1 more finely resolve topographic pattern of erosion, with the highest (red) happening on hill slopes rather than hilltops.*

When summarized over the 33 subwatersheds, the mean values of  $E$  are revealed to be within a range of -5.20 to 5.45 (Fig. 9). Parcels within subwatersheds with a mean  $E$  below -1 were counted as deserving of special attention for being in more erosive regions – these parcels constituted 24.7% of the watershed by area (counting only unprotected parcels).

## Connectivity and Corridors

Using the least-cost-distance method, highly connective patches of land were found primarily in the eastern portion of the watershed, where a large contiguous swath of forested land closely surrounded by smaller discrete patches greatly enhances the  $C$



*Fig. 8. General pattern of erosion values (0,-1) predict more pronounced erosive regions in the urban area.*

metrics found in that region. The most isolated patches are visibly small and distant from their neighbors, embedded in a matrix of urban development (Fig. 10).

It should be noted here that certain anomalies may arise in the application of the connectivity tool in its current form. These may occur when a very small patch is very close to other patches of high connectivity, but is strictly speaking not contiguous with them, and

is thus counted as a separate patch. When a parcel intersects with such a patch, it may receive a higher connectivity score than a parcel found on a very large, highly connected patch that nevertheless is sharing its  $C$  score with many other parcels. This scenario is misleading, as the former parcel and patch could very well be extremely small and hardly

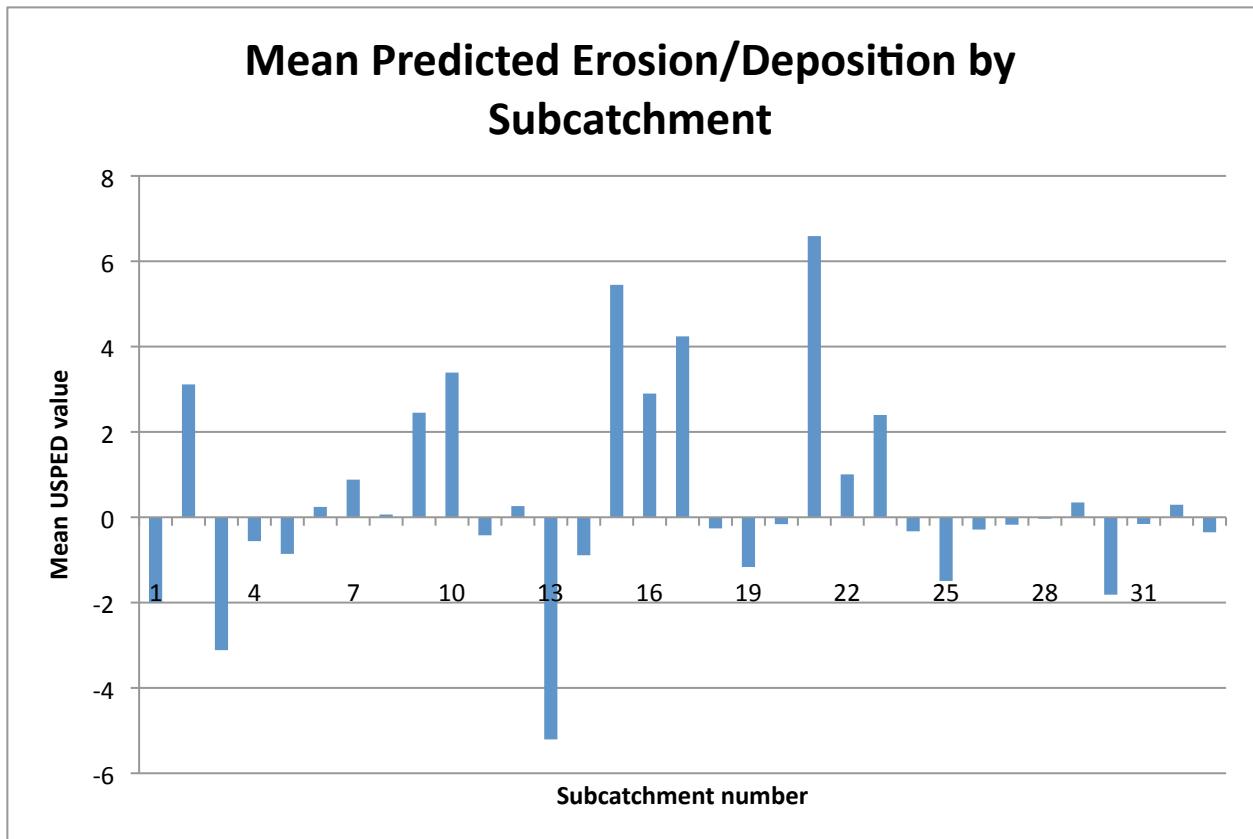
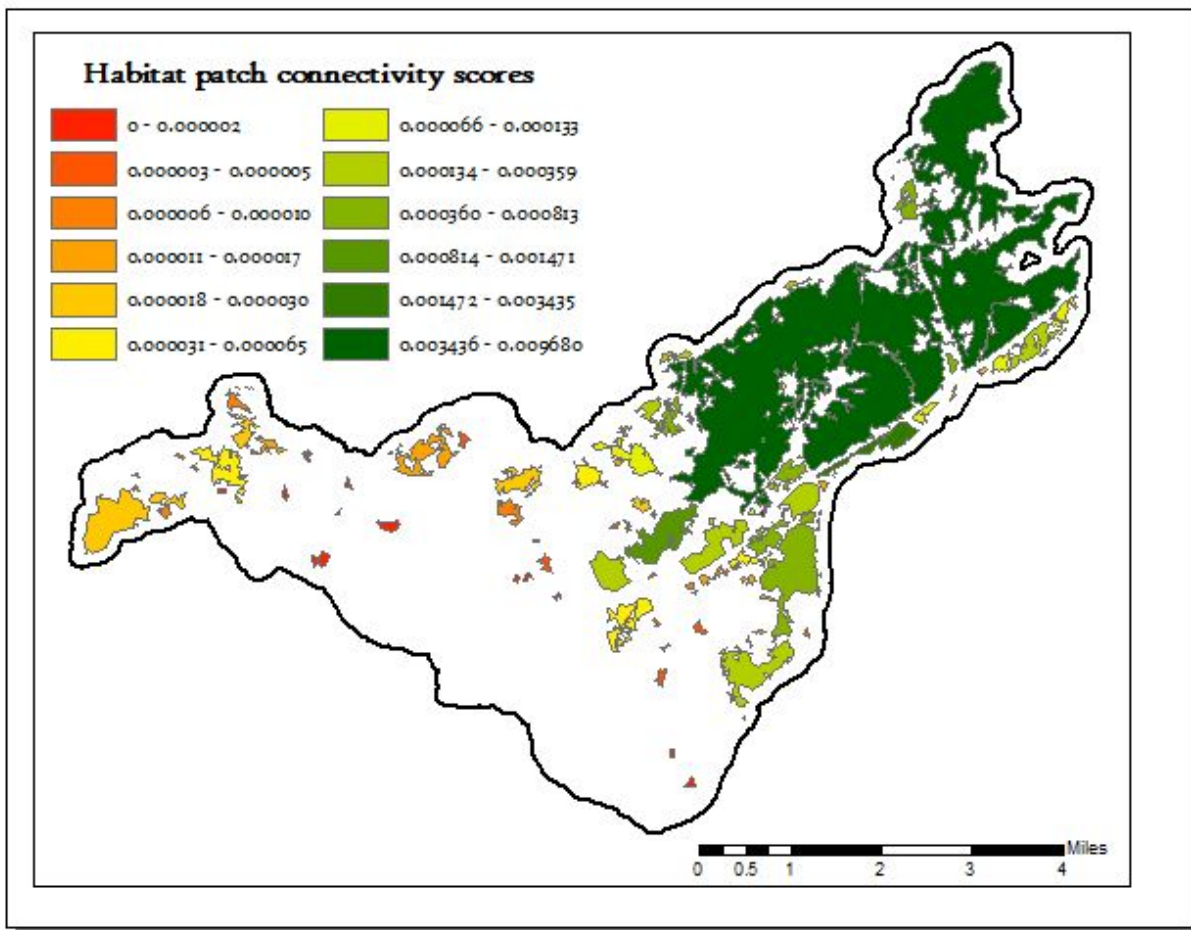


Fig. 9. Mean predicted USPED values by subwatershed, predicting subwatersheds with net erosion and net deposition.

of major significance to connectivity. This situation may be resolved by introducing a clustering algorithm similar to that employed in the underserved areas script, in which individual patches within a user-specified distance of each other are treated as one patch feature even if they are not strictly contiguous, and this feature will soon be introduced.

Corridors were produced between the 10 largest habitat patches in the watershed, using a cutoff value of 450, meaning 450 of the lowest cost-distance values were extracted

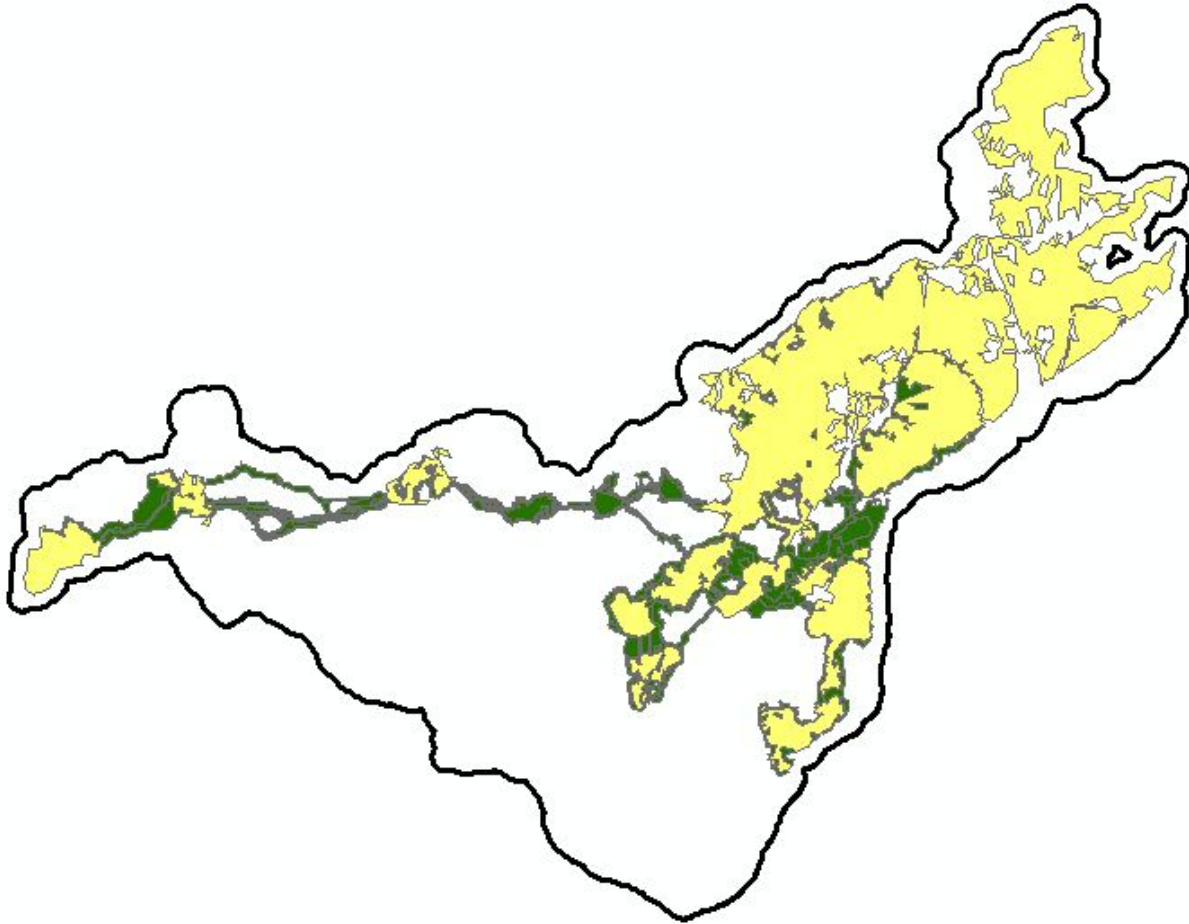
to form the corridors (Fig. 11). Many of these corridors are overlapping, as is to be expected, as a corridor connecting any two patches is itself likely to pass through one or more other patches on its way, since these, by definition, represent some of the lowest-cost matrix available. A closer examination of the results reveals that some corridors are in fact non-



*Fig. 10. Patches of forested land in Ellerbe Creek characterized by their least-cost connectivity to one another.*

contiguous slivers. Such formations generally occur between patches that are close together, and while they are not technically “corridors,” they do have a physical meaning insofar as they represent parts of the matrix around patches and inside patch gaps that

may be traversed freely without exceeding the cost threshold. These may be seen as “roaming” areas around patches or as a kind of least-cost buffer.



*Fig. 11. Least-cost corridors (green) between 10 largest forest patches in the watershed.*

### **Alternative Prioritizations and Comparison to Existing Plans**

The proportions of watershed parcels meeting each criterion may be found summarized in Table 2. The alternative prioritizations described in Table 1 were carried out to explore the effect of emphasizing different categories of criteria relative to a baseline weighting that is most likely to reflect conservation priorities in Ellerbe Creek. After top parcels were determined by sorting them in decreasing order by their final weighted score,

and within that, by their size, the percentages of the top 250 parcels meeting each criterion were calculated and may be seen in Fig. 12.

Most criteria respond as expected to increased emphasis on their category (water quality, landscape integrity, or non-ecological considerations), though it appears more

*Table 2. Parcels in watershed meeting the established criteria (blue = water quality; green = landscape integrity; pink = non-ecological considerations; orange = parcels identified in already existing analyses with similar goals).*

Criterion	% of unprotected parcels	% of unprotected parcels (by acreage)	% of all parcels	% of all parcels (by acreage)
In high-erosion subcatchment*	26.3%	24.7%	26.2%	20.7%
In subcatchment with <10% imperviousness	11.8%	27.2%	12.0%	31.6%
Within 100 ft. of Ellerbe Creek or major tributaries	2.0%	9.5%	2.5%	17.4%
In 80th percentile or higher of contributing to landscape connectivity*	2.7%	35.4%	3.0%	46.2%
In a connectivity corridor connecting 10 largest patches (excluding patches themselves)*	3.2%	5.2%	3.2%	4.2%
Contain diabase features	5.2%	16.0%	5.4%	22.0%
In an "underserved area" (50th percentile or lower of green-space service areas)*	40.3%	35.1%	40.1%	31.3%
Within 1/4 mile of a school	40.4%	27.1%	40.3%	23.0%
Within the boundaries of a neighborhood association	39.9%	25.2%	39.8%	21.7%
Identified in Watershed Management Improvement Plan (2010)	2.2%	19.3%	2.2%	17.1%
Contains proposed BMP or stream restoration project	3.2%	16.5%	3.5%	20.8%
Qualifies for funding under UNCWI analysis	5.7%	33.7%	6.4%	45.9%

parcels within 100 ft. of Ellerbe Creek are captured under the non-ecological emphasis scenario, perhaps because otherwise weakly scoring parcels within this buffer are greatly aided by having their non-ecological qualifications weighted more heavily. In terms of a weighting strategy, this suggests that a higher emphasis on non-ecological criteria will not cause any significant neglect of parcels within the 100-ft. buffer (although it will do so very much for the <10% imperviousness criterion).

Comparison of the top 250 parcels under the baseline prioritization scenario to those selected by existing analyses (Fig. 13) reveals significantly greater overlap with the

UNCWI analysis than with either the Ellerbe Creek CAPP or the parcels in the WMIP identified for BMP and/or stream restoration projects. This suggests that the criteria and the models used to evaluate them in this analysis were distinct enough from those used in the existing plans, especially the latter two, that this analysis provides a useful complement

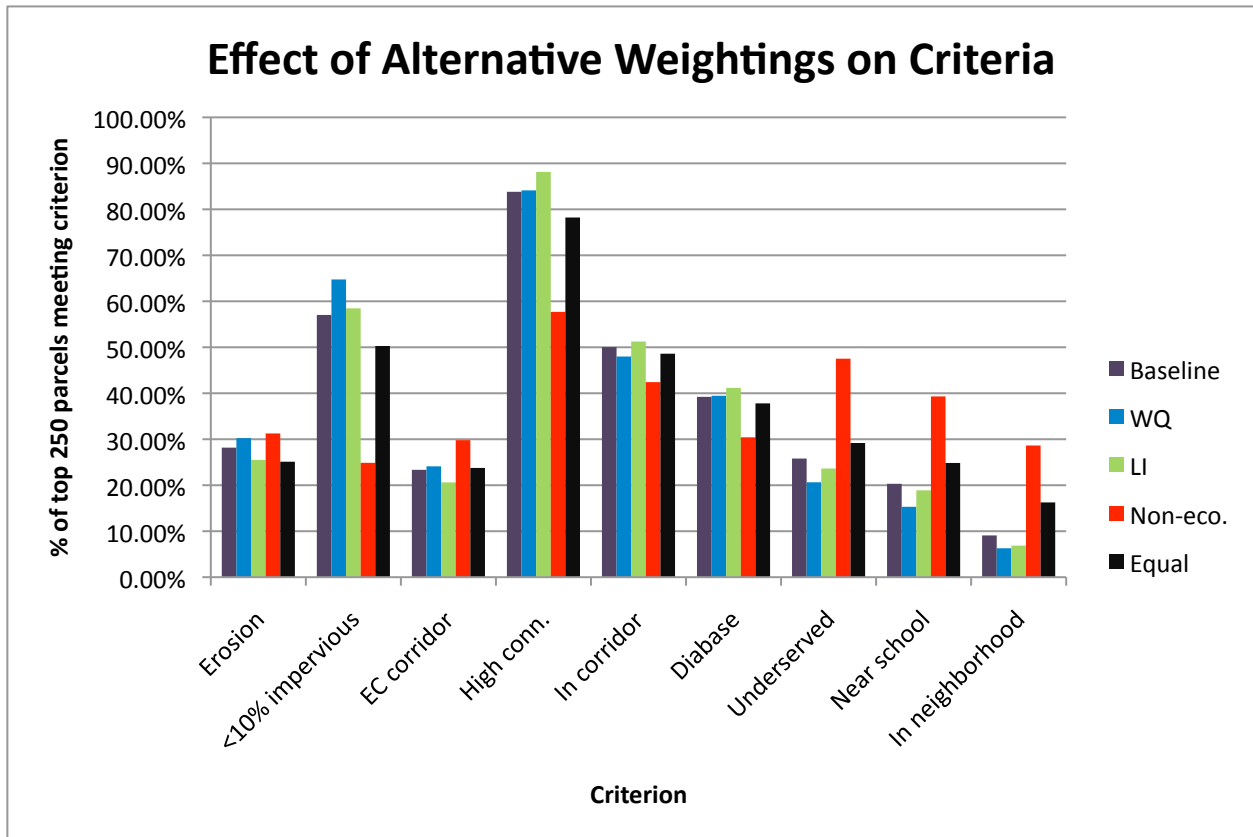


Fig. 12. Percentages of top 250 parcels meeting each criterion under each of 5 weighting scenarios.

to what has already been accomplished. Appearing in the existing plans may be incorporated into this analysis as a separate criterion without fear of creating redundancy.

### Tool Limitations and Future Work

The tool presented here depends heavily on the accuracy of the data and on user-supplied parameters. Leaving aside the issue of weightings, inputs such as the cost surface

that is supplied for the connectivity and corridor analysis, the percentile cutoff values that determine which parcels meet a criterion enough to be flagged with a point, the threshold distance for grouping open-space polygons into clusters in the underserved areas analysis, and perhaps most of all, the factors used to compute the USPED all may be expected to have a very significant impact on the ultimate conservation priorities that the tool suggests. However, the automated nature of the tool, by saving the user the prolonged work of

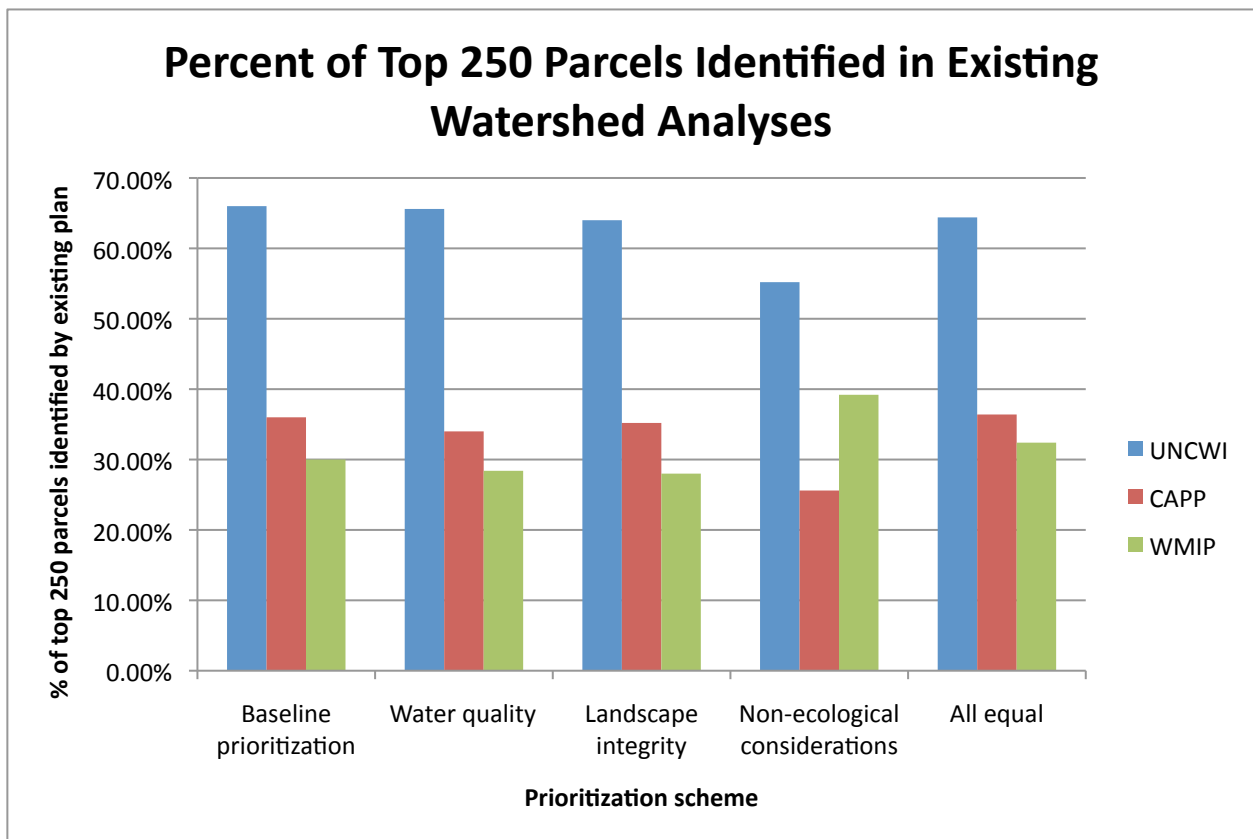


Fig. 13. Comparison of top 250 parcels identified in baseline prioritization scenario to parcels identified in existing analyses for watershed conservation/restoration.

building a geoprocessing model and then running it countless times, facilitates an increased focus on developing these high-quality inputs.

Two significant issues for watershed modeling in ArcGIS presented themselves during the construction of this tool, which prevented more thorough modeling of potential

nutrient sinks. First, in an urban setting, Ellerbe Creek included, the stream network is characterized by an overwhelming amount of sewer pipes and artificial channels that, since they are manmade, do not conform to topography as represented by a DEM. Therefore, a stream network and appropriate flow paths that account for these artificial channels are not derivable from an elevation map in the traditional way. The best way to represent a hydrologically sound stream network that includes both natural and artificial channels in a geoprocessing environment appears to be a standing research problem. In the context of the Ellerbe Creek watershed, which passes through a major urban area, a truly reliable representation of flow paths, and by extension, of nutrient transport, will not be possible without first addressing this problem.

Secondly, when employing the flow accumulation algorithm in ArcGIS to model nutrient accumulation along a flow path using a weighting raster, it would be desirable to represent nutrient losses to landscape sinks with negative values, forcing a kind of “decumulation,” in which the flow accumulation algorithm would keep a running budget of the amount of nutrient in a cell based on the sinks and sources through which it passes. A limitation of ArcGIS’s flow accumulation feature is that such negative values are treated as zeroes. Therefore, the only way to represent nutrient losses along a flow path is through proportional decay, which is inaccurate, as it is independent of the original quantity present. If this problem were to be addressed with a flow accumulation script that runs external to ArcGIS and is able to both add and subtract values, it would present a significant step forward in being able to spatially model nutrient sources and sinks for a relatively modest investment of effort.

## **Conclusion**

The tool presented here equips conservation organizations with several relatively more complex geoprocessing models for use in their planning efforts, the automated nature of the tool enabling additional resources to be devoted to developing scientifically sound inputs, additional criteria to supplement those evaluated by the tool, and the field verification of GIS data and predictions. While the tool is only a supplement to the complex, deliberative, and multi-stakeholder process that is watershed conservation planning – a process that is never truly completed – it is hoped that this tool may be extremely useful for focusing discussion and helping organizations dig deeper into the details of specific properties even as they articulate and examine their broader goals and priorities.

## References

- Baker, M.E., D.E. Weller, and T.E. Jordan. 2006. Improved methods for quantifying potential nutrient interception by riparian buffers. *Landscape Ecology* 21: 1237-1345.
- Beier, P. and R.F. Noss. 1998. Do habitat corridors provide connectivity? *Conservation Biology* 12(6): 1241-1252.
- Bernhardt, E.S. and M.A. Palmer, 2007. Restoring streams in an urbanizing world. *Freshwater Biology* 52: 738-751.
- Booth, D.B. and C.R. Jackson. 1997. Urbanization of aquatic systems: Degradation thresholds, stormwater detection, and the limits of mitigation. *Journal of the American Water Resources Association* 33(5): 1077-1090.
- Brown and Caldwell. 2010. Ellerbe Creek Watershed Management Improvement Plan. Volume II: Watershed Management Improvement Plan and Appendix M: Critical Areas Protection Plan.
- Brown, L.C. and G.R. Foster. 1987. Storm erosivity using idealized intensity distributions. *Transactions of the American Society of Agricultural Engineers* 30: 379-386.
- Brown, J.H., V.K. Gupta, B. Li, B.T. Milne, C. Restrep, and G.B. West. 2002. The fractal nature of nature: Power laws, ecological complexity, and biodiversity. *Philosophical Transactions of the Royal Society of London B: Biological Sciences* 357: 619-626.
- Calabrese, J.M. and W.F. Fagan. 2004. A comparison-shopper's guide to connectivity metrics. *Frontiers in Ecology and the Environment* 2: 529-536.
- Crooks, K.R. 2002. Relative sensitivities of mammalian carnivores to habitat fragmentation. *Conservation Biology* 16: 1-15.
- Karpilo, Jr., R.D. and T.J. Toy. 2003. Non-agricultural C and P values for RUSLE. *Proceedings from the National Meetings of the American Society of Mining and Reclamation (AMSR), Joint Conference of the 9<sup>th</sup> Billings Land Reclamation Symposium and the 20<sup>th</sup> Annual Meetings of the American Society of Mining and Reclamation, Billings, MT.*
- Klein, R.D. 1979. Urbanization and stream quality impairment. *Water Resources Bulletin* 15(4): 948-963.
- Lachman, B.E., A. Wong, and S.A. Resetar. 2007. The Thin Green Line: An Assessment of DoD's Readiness and Environmental Protection Initiative to Buffer Installation Encroachment. RAND Corporation, Santa Monica, CA.
- Leh, M., S. Balwa, and I. Chaubey. 2011. Impact of land use change on erosion risk: An integrated remote sensing, geographic information system, and modeling methodology.

*Land Degradation and Development*, published online in Wiley Online Library (wileyonlinelibrary.com) DOI: 10.1002/ldr.1137.

Leopold, L.B., R. Huppman, and A. Miller. 2005. Geomorphic effects of urbanization in forty-one years of observation. *Proceedings of the American Philosophical Society* 149: 349-371.

Magle, S.B., D.M. Theobald, and K.R. Crooks. 2009. A comparison of metrics predicting landscape connectivity for a highly interactive species along an urban gradient in Colorado, USA. *Landscape Ecology* 24: 267-280.

Mitas, L. and H. Mitasova. 1998. Erosion/deposition modeling with USPED using GIS. Available online at: <http://skagit.meas.ncsu.edu/~helena/gmslab/erosion/usped.html>

Mitasova, H., J. Hofierka, M. Zlocha, and L.R. Iverson. 1996. Modeling topographic potential for erosion and deposition using GIS. *International Journal of Geographic Information Science* 10: 629-641.

Moilanen, A. and I. Hanski. 2001. On the use of connectivity measures in spatial ecology. *Oikos* 95: 147-151.

Moore, I. and G. Burch. 1986. Physical basis of the length-slope factor in the Universal Soil Loss Equation. *Soil Science Society of America Journal* 50: 1294-1298.

Morgan, R.P. and S.E. Cushman. 2005. Urbanization effects on stream fish assemblages in Maryland, USA. *Journal of the North American Benthological Society* 24: 643-655.

North Carolina Department of Environmental and Natural Resources (NCDENR), Division of Water Quality. 2001. The Qualitative 5-Sample Procedure. Raleigh, NC.

North Carolina Ecosystem Enhancement Program (NCEEP). 2006. Ellerbe Creek Local Watershed Plan. Available at: [http://www.nceep.net/services/wps/Upper\\_Neuse/Ellerbe\\_Creek\\_Local\\_Watershed\\_Plan.pdf](http://www.nceep.net/services/wps/Upper_Neuse/Ellerbe_Creek_Local_Watershed_Plan.pdf)

North Carolina General Statutes, Chapter 113A, Article 18. Clean Water Management Trust Fund.

Toy, T.J., G.R. Foster, and K.G. Renard. 1999. RUSLE for mining, construction, and reclamation lands. *Journal of Soil and Water Conservation* 54(2): 462-467.

The Trust for Public Land. 2005. *Source Protection Handbook: Using Land Conservation to Protect Drinking Water Supplies*. Trust for Public Land and American Water Works Association.

The Trust for Public Land. 2006. The Upper Neuse Clean Water Initiative Conservation Plan: Protecting Land and Drinking Water for the Future.

UNC Environmental Finance Center. 2012. Upper Neuse River Basin Watershed Protection Revenueshed Analysis. School of Government, University of North Carolina at Chapel Hill, Chapel Hill, NC.

Walker, A.S. 2009. Soil seedbank of rare plant communities associated with diabase soils in Durham and Granville Counties, North Carolina. M.S. Botany Thesis, North Carolina State University.

Wischmeier, W.H. and D.D. Smith. 1978. Predicting Rainfall Erosion Losses: A Guide to Conservation Planning. Agriculture Handbook No. 537. USDA-Agricultural Research Service: Washington, DC.

## Appendix: Python Codes

### Euclidean-distance connectivity

```
import sys, arcpy
from arcpy import env
from arcpy import sa
```

```
arcpy.CheckOutExtension("spatial")
arcpy.env.overwriteOutput = True
```

```
landuse = sys.argv[1]
dispersal = sys.argv[2]
parcels = sys.argv[3]
P = sys.argv[4]
workspace = sys.argv[5]
arcpy.env.workspace = workspace
arcpy.env.snapRaster = landuse
arcpy.env.extent = landuse
arcpy.env.cellSize = landuse
```

```
arcpy.AddWarning("Extracting natural land covers...") #Get natural habitats of interest out of NLCD data
```

```
NatLands = arcpy.sa.Reclassify(landuse, "Value", "11 11;21 NODATA;22 NODATA;23 NODATA;24 NODATA;31 NODATA;41 41;42 42;43 43;52 52;71 71;81 NODATA;82 NODATA;90 90;95 95", "DATA")
```

```
arcpy.AddWarning("Grouping patches...") #Group contiguous habitat cells into discrete patches
```

```
NatLandsGrp = arcpy.sa.RegionGroup(NatLands, "EIGHT", "CROSS", "NO_LINK", "")
```

```
arcpy.AddWarning("Converting to polygons...") #Convert discrete patches to polygons
```

```
NatPoly =
arcpy.RasterToPolygon_conversion(NatLandsGrp, "PatchPoly.shp", "SIMPLIFY", "Value")
```

**#Polygon conversion causes slivers of polygons to be counted as individual features when they are in fact contiguous with a large adjacent polygon. These slivers share a GRIDCODE attribute with the polygon they're contiguous with, as they were originally both the same patch. Dissolving by GRIDCODE re-lumps these slivers back with those polygons.**

```
arcpy.AddWarning("Consolidating erroneous slivers...")
```

```
NatPoly2 =
arcpy.Dissolve_management(NatPoly, "patches.shp", "GRIDCODE", "", "MULTI_PART", "DISSOLVE_LINES")
```

arcpy.AddWarning("Calculating patch areas...") **#Calculates areas of patch polygons, in acres**

```
arcpy.AddField_management(NatPoly2,"Area",  
"FLOAT","","","","NON_NULLABLE","NON_REQUIRED","")  
arcpy.CalculateField_management(NatPoly2,"Area","!shape.area@ACRES!","PYTHON_9.3",  
")
```

**#Generates a table giving the distance of each patch to every other patch within the user-specified "dispersal" distance**

```
arcpy.AddWarning("Calculating distances between patches...")  
NearTable =  
arcpy.GenerateNearTable_analysis(NatPoly2,NatPoly2,"NearTable",dispersal,"NO_LOCATION",  
"NO_ANGLE","ALL")
```

**#Adds one field giving the area of the "focal" patch (IN\_FID) and another giving the area of the patch the distance to which was measured (NEAR\_FID) in the given row**

```
arcpy.AddWarning("Adding areas to table of distances...")  
arcpy.JoinField_management(NearTable,"IN_FID",NatPoly2,"FID","Area")  
arcpy.JoinField_management(NearTable,"NEAR_FID",NatPoly2,"FID","Area")
```

**#Adds new field to contain connectivity/isolation score. Calculates this as Score = (Area1 \* Area2)^0.7 / (distance b/w the patches)^1.7**

```
arcpy.AddWarning("Calculating connectivity score for each patch pair...")  
arcpy.AddField_management(NearTable,"Connect","FLOAT","","","NON_NULLABLE",  
NON_REQUIRED,"")  
arcpy.CalculateField_management(NearTable,"CONNECT","(( !AREA! * !AREA_1!) ** 0.7 /  
!NEAR_DIST! ** 1.7) * 100","PYTHON_9.3","")
```

**#At this point, table is listing connectivity score between each patch and each other patch. This process consolidates these scores for each "focal" patch by summing them.**

```
arcpy.AddWarning("Consolidating connectivity scores...")  
arcpy.Statistics_analysis(NearTable,"NearTabStats","CONNECT SUM","IN_FID")
```

**#Connectivity scores are joined to the patch polygons.**

```
arcpy.AddWarning("Joining connectivity scores to patches...")  
arcpy.JoinField_management(NatPoly2,"FID","NearTabStats","IN_FID","SUM_CONNECT")
```

**#A feature layer of the patches is created with the "Use ratio policy" enabled for the connectivity score attribute. The Identity tool then computes the geometric intersection of the patches and user-supplied parcel features. Because the "use ratio policy" was used, the connectivity score is split in proportion to the amount of the patch that is actually found on a given parcel. Thus, the connectivity score, in being associated with parcel features, is effectively "weighted" by the proportion of the patch that is actually on the parcel. A Dissolve operation is used to total the connectivity scores associated with any given parcel.**

```

arcpy.AddWarning("Modifying connectivity scores based on parcels...")
arcpy.MakeFeatureLayer_management(NatPoly2,"NatPolyLyr","","","FID FID VISIBLE
NONE;Shape Shape HIDDEN NONE;GRIDCODE GRIDCODE HIDDEN NONE;Area Area
HIDDEN NONE;SUM_CONNEC SUM_CONNEC VISIBLE RATIO")
PatchIden =
arcpy.Identity_analysis("NatPolyLyr",parcels,"PatchIden.shp","ONLY_FID","","NO_RELATIO
NSHIPS")
fields = []
fieldList = arcpy.ListFields(PatchIden)
for field in fieldList:
    fields.append(field.name)
parcelField = str(fields[4])
PatchIdenDiss =
arcpy.Dissolve_management(PatchIden,"PatchIdenDiss.shp",parcelField,"SUM_CONNEC
SUM","MULTI_PART","DISSOLVE_LINES")

```

**#In order to run the tool multiple times, any fields created in previous runs are deleted from the parcels data.**

```

arcpy.AddWarning("Removing any fields from previous runs...")
fields = [] #Creates empty field list
fieldList = arcpy.ListFields(parcels) #Gets list of fields from parcels shapefile
for field in fieldList:
    fields.append(field.name) #Gets name of field from fieldList and appends it to the
empty list created above
if u'SUM_SUM_CO' in fields:
    arcpy.DeleteField_management(parcels,"SUM_SUM_CO") #Looks if field containing
connectivity score exists in the list -- if so, deletes it from parcels shapefile
if u'ConnPt' in fields:
    arcpy.DeleteField_management(parcels,"ConnPt") #Looks if field containing parcel
"point" exists in the list -- if so, deletes it from parcels shapefile

```

**#Parcel-scale connectivity scores are joined to the parcel polygons -- this can be time-consuming, lasting 7-10 minutes for the trial dataset (which contains 138 patches and ~20,000 parcels)**

```

arcpy.AddWarning("Joining final scores to parcels data...")
arcpy.JoinField_management(parcels,"FID",PatchIdenDiss,parcelField,"SUM_SUM_CO")

```

```

arcpy.AddWarning("Awarding 1 point to parcels with higher connectivity...")
connList = [] #Creates empty list
where = "SUM_SUM_CO" > 0 #Creates where clause for subsequent search cursor,
specifying connectivity scores above 0
rows = arcpy.SearchCursor(parcels,where","","SUM_SUM_CO","SUM_SUM_CO A") #Activates
search cursor to look through connectivity attribute (which is sorted in ascending
order)
for row in rows:

```

```
connList.append(row.SUM_SUM_CO) #Appends connectivity scores above to a list in
ascending order
del row, rows
```

**#Defines function that will return the value from a list that is at a user-specified percentile**

```
def percentile(connList,P):
    n = int(round(P * len(connList) + 0.5))
    if n > 1:
        return connList[n-2]
    else:
        return 0
```

**#"Cutoff" value is the value returned by the percentile function ... anything above this value can be considered to be a "high" connectivity score**

```
cutoff = percentile(connList,float(P))
```

**#Add a field to parcels data ... if the connectivity score on the parcel is above the user-specified cutoff percentile, use an update cursor to award that parcel a "1" (a point)**

```
arcpy.AddField_management(parcels,"ConnPt","FLOAT","","","","NON_NULLABLE","NON_
REQUIRED","")
cutoffClause = "SUM_SUM_CO > " + str(cutoff)
rows = arcpy.UpdateCursor(parcels,cutoffClause,"","ConnPt")
for row in rows:
    row.ConnPt = 1
    rows.updateRow(row)
del row, rows
```

**#Delete unnecessary intermediates**

```
arcpy.AddWarning("Deleting intermediates...")
arcpy.Delete_management("PatchPoly.shp")
arcpy.Delete_management(NatLands)
arcpy.Delete_management(NearTable)
arcpy.Delete_management("NearTabStats")
arcpy.Delete_management("NatPolyLyr")
arcpy.Delete_management(PatchIden)
arcpy.Delete_management(PatchIdenDiss)
```

### **Least-cost connectivity**

```
import os, sys, arcpy
from arcpy import env
from arcpy import sa

arcpy.CheckOutExtension("spatial")

landuses = sys.argv[1]
costsurf = sys.argv[2]
parcels = sys.argv[3]
P = sys.argv[4]
workspace = sys.argv[5]

arcpy.env.overwriteOutput = True
arcpy.env.workspace = workspace
arcpy.env.snapRaster = landuses
arcpy.env.extent = landuses
arcpy.env.cellSize = landuses
spatialRef = arcpy.Describe(landuses).spatialReference
arcpy.env.outputCoordinateSystem = spatialRef

arcpy.AddWarning("Creating discrete habitat patches...")
patches = arcpy.sa.RegionGroup(landuses,"EIGHT","CROSS","NO_LINK","")
patches.save(os.path.join(workspace,"patchrast"))

arcpy.AddWarning("Creating list of patches in memory...")
patchNums = []
rows = arcpy.SearchCursor(patches)
for row in rows:
    patchNums.append(row.Value)
del row, rows

arcpy.AddWarning("Creating empty .csv file to record cost distances...")
distList = os.path.join(workspace,"distlist.csv")
distTable = open(distList,'w')
distTable.write("FocalPatch,ToPatch,CostDist\n")

arcpy.AddWarning("Iteratively recording least-cost distance from each patch to every
other patch...")
for patchNum in patchNums:
    focalPatch = arcpy.sa.SetNull(patches,patches,"VALUE <> " + str(patchNum))
    costDist = arcpy.sa.CostDistance(focalPatch,costsurf)
    zonalStats =
arcpy.sa.ZonalStatisticsAsTable(patches,"Value",costDist,"in_memory\costMin","DATA","M
INIMUM")
    rows = arcpy.SearchCursor(zonalStats)
```

```

row = rows.next()
while row:
    distTable.write("%d,%d,%d\n" %(patchNum,row.VALUE,row.MIN))
    row = rows.next()
del row, rows

distTable.close()

arcpy.AddWarning("Converting cost-distance table to DBF format..")
arcpy.CopyRows_management(distList,"distTableDBF.dbf")

arcpy.AddWarning("Converting patches to polygons...")
arcpy.RasterToPolygon_conversion("patchrast","PatchPoly.shp","SIMPLIFY","Value")

arcpy.AddWarning("Consolidating erroneous slivers...")
arcpy.Dissolve_management("PatchPoly.shp","patches.shp","GRIDCODE","","MULTI_PART",
"DISSOLVE_LINES")

arcpy.AddWarning("Calculating patch areas...")
arcpy.AddField_management("patches.shp","Area","FLOAT","","","","NON_NULLABLE","N
ON_REQUIRED","")
arcpy.CalculateField_management("patches.shp","Area","!shape.area@ACRES!","PYTHON_9
.3","")

arcpy.AddWarning("Adding areas to table of distances...")
arcpy.JoinField_management("distTableDBF.dbf","FocalPatch","patches.shp","GRIDCODE","
Area")
arcpy.JoinField_management("distTableDBF.dbf","ToPatch","patches.shp","GRIDCODE","Ar
ea")

arcpy.AddWarning("Calculating connectivity score for each patch pair...")
arcpy.AddField_management("distTableDBF.dbf","Connect","FLOAT","","","","NON_NULL
ABLE","NON_REQUIRED","")
arcpy.CalculateField_management("distTableDBF.dbf","CONNECT","(!AREA! * !AREA_1!)
** 0.7 / !COSTDIST! ** 1.7) * 100","PYTHON_9.3","")

arcpy.AddWarning("Consolidating connectivity scores...")
arcpy.Statistics_analysis("distTableDBF.dbf","DistTabSum",[["Connect","SUM]],"FocalPatc
h")

arcpy.AddWarning("Joining connectivity scores to patches...")
arcpy.JoinField_management("patches.shp","GRIDCODE","DistTabSum","FOCALPATCH","S
UM_CONNECT")

arcpy.AddWarning("Modifying connectivity scores based on parcels...")

```

```
arcpy.MakeFeatureLayer_management("patches.shp","PatchLyr","","","FID FID VISIBLE NONE;Shape Shape HIDDEN NONE;GRIDCODE GRIDCODE HIDDEN NONE;Area Area HIDDEN NONE;SUM_CONNEC SUM_CONNEC VISIBLE RATIO")
```

```
arcpy.Identity_analysis("PatchLyr",parcels,"PatchIden.shp","ONLY_FID","","NO_RELATIONS HIPS")
```

```
fieldList = arcpy.ListFields("PatchIden.shp")  
dissField = str(fieldList[4].name)  
arcpy.Dissolve_management("PatchIden.shp","PatchIdenDiss.shp",dissField,"SUM_CONNEC SUM","MULTI_PART","DISSOLVE_LINES")
```

```
arcpy.AddWarning("Removing any fields from previous runs...")
```

```
fields = []  
fieldList2 = arcpy.ListFields(parcels)  
for field in fieldList2:  
    fields.append(field.name)  
if u'SUM_SUM_CO' in fields:  
    arcpy.DeleteField_management(parcels,"SUM_SUM_CO")  
if u'ConnPt' in fields:  
    arcpy.DeleteField_management(parcels,"ConnPt")
```

```
arcpy.AddWarning("Joining final scores to parcels data...")  
arcpy.JoinField_management(parcels,"FID","PatchIdenDiss.shp",dissField,"SUM_SUM_CO")
```

```
arcpy.AddWarning("Awarding 1 point to parcels with higher connectivity...")  
connList = []  
where = "SUM_SUM_CO" > 0'  
rows = arcpy.SearchCursor(parcels,where,"","SUM_SUM_CO","SUM_SUM_CO A")  
for row in rows:  
    connList.append(row.SUM_SUM_CO)  
del row, rows
```

```
def percentile(connList,P):  
    n = int(round(P * len(connList) + 0.5))  
    if n > 1:  
        return connList[n-2]  
    else:  
        return 0
```

```
cutoff = percentile(connList,float(P))
```

```
arcpy.AddField_management(parcels,"ConnPt","FLOAT","","","NON_NULLABLE","NON_REQUIRED","")  
cutoffClause = "SUM_SUM_CO > " + str(cutoff)  
rows = arcpy.UpdateCursor(parcels,cutoffClause,"","ConnPt")
```

```
for row in rows:
    row.ConnPt = 1
    rows.updateRow(row)
del row, rows

arcpy.AddWarning("Deleting intermediates...")
arcpy.Delete_management(distList)
arcpy.Delete_management("distTableDBF.dbf")
arcpy.Delete_management("DistTabSum")
arcpy.Delete_management("PatchIden.shp")
arcpy.Delete_management("PatchIdenDiss.shp")
arcpy.Delete_management("patchpoly.shp")
arcpy.Delete_management("patchrast")
```

## **Corridor generation**

```
import os, sys, arcpy, fnmatch, itertools
from arcpy import env
from arcpy import sa
```

```
arcpy.CheckOutExtension("spatial")
```

```
topPatch = sys.argv[1] #User supplies patches to consider ...
costsurf = sys.argv[2] #...cost surface...
corrCutoff = sys.argv[3] #...corridor restrictiveness threshold...
corridors = sys.argv[4] #...name for new shapefile that will hold corridors...
parcels = sys.argv[5] #...parcels shapefile...
workspace = sys.argv[6] #...and a directory to send the outputs.
```

```
arcpy.env.overwriteOutput = True
arcpy.env.workspace = workspace
arcpy.env.snapRaster = costsurf
arcpy.env.extent = costsurf
arcpy.env.cellSize = costsurf
spatialRef = arcpy.Describe(topPatch).spatialReference
arcpy.env.outputCoordinateSystem = spatialRef
```

```
description = arcpy.Describe(costsurf)
cellSize = description.children[0].meanCellHeight
```

```
arcpy.PolygonToRaster_conversion(topPatch,"GRIDCODE","CorrPatches","CELL_CENTER",
",cellSize)
```

```
arcpy.AddWarning("Creating new directories...")
distanceDir = arcpy.CreateFolder_management(workspace,"CostDists") #To contain cost-
distance rasters
distDirString = str(distanceDir)
corridorDir = arcpy.CreateFolder_management(workspace,"Corridors") #To contain
corridor shapefile
corrDirString = str(corridorDir)
```

```
arcpy.AddWarning("Creating empty feature class to contain corridors...")
arcpy.CreateFeatureclass_management(os.path.dirname(corridors),os.path.basename(corr
idors),"POLYGON","", "ENABLED","",spatialRef)
arcpy.AddField_management(corridors,"From","LONG")
arcpy.AddField_management(corridors,"To","LONG")
```

```
arcpy.AddWarning("Creating list of patches in memory...")
patchNums = []
rows = arcpy.SearchCursor("CorrPatches")
for row in rows:
```

```
    patchNums.append(row.Value)
del row, rows
```

```
arcpy.AddWarning("Generating cost distance rasters for each patch...")
```

```
for patchNum in patchNums:
```

```
    focalPatch = arcpy.sa.SetNull("CorrPatches","CorrPatches","VALUE <> " +
str(patchNum)) #If value (patch number) doesn't equal x, set null; if value does equal
x (if "condition is false"), keep it; i.e. only patch of interest gets pulled out
```

```
    costDist = arcpy.sa.CostDistance(focalPatch,costsurf) #Get cost distance surface to that
patch
```

```
    withhole = arcpy.sa.SetNull(~ arcpy.sa.IsNull(focalPatch),costDist) #Where focal patch
is not null (where it exists), set that part of the cost distance raster to null (i.e. pokes
a hole in it)
```

```
    withhole.save(os.path.join(distDirString,"dist_%s.img" %patchNum)) #Save cost-
distance raster, w/ hole where destination patch is, to CostDists folder; include
number of patch the loop is on in the filename
```

```
withhole = None #Necessary to fully execute last iteration of loop
```

```
arcpy.AddWarning("Generating corridor rasters...")
```

```
newDir = workspace + "CostDists/" #Make path to subfolder storing cost distances
rasters.
```

```
costRasts = os.listdir(newDir) #List all files in this subfolder.
```

```
costRasts2 = fnmatch.filter(costRasts,"*.img") #Filter the list to exclude the .aux.xml
auxiliary files.
```

```
rastPaths = [] #Make empty list.
```

```
for Rast in costRasts2: #Begin loop through list of cost distance rasters in their
subfolder...
```

```
    fullPath = os.path.join(newDir,Rast) #Build string pointing to the specific cost
distance raster file...
```

```
    rastPaths.append(fullPath) #Append that string to a list.
```

```
combos = itertools.combinations(rastPaths,2) #Create combinations object pairing all
possible combinations of cost distance raster files to be used in Corridor, w/o
repeating any pairings.
```

```
comboList = list(combos) #Convert combinations object to a list of tuples, where each
tuple is one of these pairs.
```

```
for combo in comboList: #Loop through list of tuples...
```

```
    patch1 = combo[0] #Pull out first member of tuple (one patch)...
```

```
    patch2 = combo[1] #Pull out second member of tuple (another patch)...
```

```
    corridor = arcpy.sa.Corridor(patch1,patch2) #Create corridor raster between them...
```

```
    fromPatch = int(patch1[patch1.find("_")+1:patch1.find(".")]) #Pull out patch number
```

```
    toPatch = int(patch2[patch2.find("_")+1:patch2.find(".")]) #Pull out patch number
```

```
    corrInt = arcpy.sa.Int(corridor) #Convert raster to integer format
```

```
    corrInt.save(os.path.join(workspace,"corridor2.img"))
```

```
    corrInt = os.path.join(workspace,"corridor2.img")
```

```

corrAtt = arcpy.BuildRasterAttributeTable_management(corrInt) #Build attribute table
rasterObj = arcpy.Raster(corrInt)
corrThresh = arcpy.sa.Con(rasterObj,rasterObj,"","OID <= " + str(corrCutoff)) #Pull
corridor out of corridor raster according to restrictiveness threshold
corrFeature = "in_memory/corrFeature"
corrDissolve = "in_memory/corrDissolve"
corrPoly = arcpy.RasterToPolygon_conversion(corrThresh,corrFeature)
corrDiss = arcpy.Dissolve_management(corrFeature,corrDissolve) #Dissolve corridor
polygon to eliminate cell boundaries
cursor = arcpy.InsertCursor(corridors) #Activate cursor
feature = cursor.newRow()
feature.shape = arcpy.SearchCursor(corrDissolve).next().shape #Get feature shape
feature.From_ = fromPatch #Get number of "from" patch loop is currently on
feature.To = toPatch #Get number of "to" patch loop is currently on
cursor.insertRow(feature) #Write newly formed corridor to empty shapefile as one
feature
del feature,cursor

fields = []
fieldList = arcpy.ListFields(parcels)
for field in fieldList:
    fields.append(field.name)
if 'CorrPt' in fields:
    arcpy.DeleteField_management(parcels,"CorrPt")

arcpy.AddWarning("Awarding 1 point to parcels in a connectivity corridor...")
arcpy.AddField_management(parcels,"CorrPt","FLOAT",,"","","NON_NULLABLE","NON_
REQUIRED",)

arcpy.MakeFeatureLayer_management(parcels,"parcelLyr")
selParcels =
arcpy.SelectLayerByLocation_management("parcelLyr","HAVE_THEIR_CENTER_IN",corrido
rs)

rows = arcpy.UpdateCursor(selParcels,"","CorrPt")
for row in rows:
    row.CorrPt = 1
    rows.updateRow(row)
del row, rows

```

## Underserved areas

```
import os, sys, arcpy, fnmatch
from arcpy import env
from arcpy import sa
```

```
arcpy.CheckOutExtension("spatial")
```

```
openSpace = sys.argv[1] #User supplies open-space shapefile...
clustDist = sys.argv[2] #...clustering threshold distance to group individual features...
shedBound = sys.argv[3] #...overall target area boundary...
popBlocks = sys.argv[4] #...census blocks shapefile with population attribute...
parcels = sys.argv[5] #...parcels shapefile...
P = sys.argv[6] #...percentile cutoff for what counts as "underserved"...
workspace = sys.argv[7] #...and directory to contain outputs.
```

```
arcpy.env.overwriteOutput = True
arcpy.env.workspace = workspace
spatialRef = arcpy.Describe(openSpace).spatialReference
arcpy.env.outputCoordinateSystem = spatialRef
```

### **#Create folder to contain service-area polygons**

```
underservDir = arcpy.CreateFolder_management(workspace,"UnderservedAreas")
dirString = str(underservDir)
```

### **#Calculates distances between all features in open-space shapefile, deletes redundant pairs from the resulting table**

```
arcpy.AddWarning("Calculating distances between green space patches...")
tableOut = os.path.join(dirString,"NearTable")
nearTable =
arcpy.GenerateNearTable_analysis(openSpace,openSpace,tableOut,"","NO_LOCATION","NO_ANGLE","ALL")
nearTable2 = arcpy.DeleteIdentical_management(nearTable,"NEAR_DIST")
```

```
arcpy.AddWarning("Executing clustering algorithm...")
```

```
inFIDs = [] #Creates empty list
```

```
nearFIDs = [] #Creates empty list
```

```
rows = arcpy.SearchCursor(nearTable2,"NEAR_DIST <= " + str(clustDist))
```

```
for row in rows:
```

```
    inFIDs.append(row.IN_FID) #Creates list of park IDs on left side of near table
```

```
    nearFIDs.append(row.NEAR_FID) #Creates list of park IDs on right side of near table
```

```
del row, rows
```

```
allClusters = [] #Creates empty list to store clusters that will be created
```

### **#Turns lists to sets to make "intersection" operations possible**

```
setInFIDs = set(inFIDs)
```

```

setNearFIDs = set(nearFIDs)
inBoth = setInFIDs.intersection(setNearFIDs) #Finds FIDs that are in both "in" & "near" lists

if len(inBoth) > 0:
    for FID in inBoth: #Loops through each park ID that's on both sides of near table
        firstIndex = inFIDs.index(FID) #Returns index position in 1st list of FID that's in both lists
        firstCounterpart = nearFIDs[firstIndex] #Returns value from 2nd list that corresponds to the instance of FID-in-both-lists that's in 1st list
        secondIndex = nearFIDs.index(FID) #Returns index position in 2nd list of FID that's in both lists
        secondCounterpart = inFIDs[secondIndex] #Returns value from 1st list that corresponds to the instance of FID-in-both-lists that's in 2nd list
        cluster = [FID,firstCounterpart,secondCounterpart] #Creates list of clustered polygon FIDs (b/c if A is near B and A is near C (where "A" is the item found in both lists), then B is near C)
        allClusters.append(cluster) #Appends cluster to a master list of clusters (unmerged at this point)

def dupes(FIDList):
    found = set()
    addNum = found.add
    found2 = set(x for x in FIDList if x in found or addNum(x))
    return list(found2) #Defines function that picks out any members of a list which occur more than once.

inFIDDupes = dupes(inFIDs) #Pick out any multiple occurrences in list of in-FIDs
nearFIDDupes = dupes(nearFIDs) #Pick out any multiple occurrences in list of near-FIDs
if len(inFIDDupes) > 0: #If any multiple occurrences were found in in-FIDs...
    for dupe in inFIDDupes: #Then for each of these...
        dupeCounterparts = []
        dupeIndices = [item for item in range(len(inFIDs)) if inFIDs[item] == dupe] #Get their index positions in in-FIDs list
        for index in dupeIndices: #Loop through these index positions...
            dupeCounterpart = nearFIDs[index] #For each one, get the corresponding value from the other (near-FIDs) list
            dupeCounterparts.append(dupeCounterpart) #Add those to a list
            cluster2 = dupeCounterparts + [dupe] #The duplicate value & its counterparts all belong to a cluster (again by the logic of if A is near B & A is near C, then B is "near" C)
            allClusters.append(cluster2) #Add this cluster to our master list of clusters (unmerged at this point)
            del dupeCounterparts #Clears list of values from the other (near-FIDs) list corresponding to the duplicates in the in-FIDs list
if len(nearFIDDupes) > 0:

```

```

for dupe in nearFIDDupes: #Repeat same thing for duplicates on right side of near
table
    dupeCounterparts2 = []
    dupeIndices = [item for item in range(len(nearFIDs)) if nearFIDs[item] == dupe]
    for index in dupeIndices:
        dupeCounterpart = inFIDs[index]
        dupeCounterparts2.append(dupeCounterpart)
    cluster3 = dupeCounterparts2 + [dupe]
    allClusters.append(cluster3)
    del dupeCounterparts2

def merge(clusters):
    sets = [set(cluster) for cluster in clusters] #Make all sublists in list into sets
    i = 0
    while i < len(sets): #the length of "sets" is the number of clusters to be considered...
        j = i+1
        while j < len(sets): #i & j are counters saying "begin w/ 1st set (i=0), compare it w/
2, 3, & 4 (j), then move to 2nd set (i=1), compare it w/ 3 & 4 (j=i+1, j=i+1 again), & so
on
            if len(sets[i].intersection(sets[j])) > 0: #If the compared sets have any elements in
common, i.e. a reason for merging them...
                sets[i] = sets[i].union(sets[j]) #Combine them into one set and make that the i-
th member of "sets"
                sets.pop(j) #Remove the j-th set, which just got merged into what's now the
i-th set
            else:
                j += 1 #If the compared sets have no elements in common, look at the next
one down the line...
        i += 1 #When done with all that, add 1 to the counter i, & repeat it all again
    merged = [list(s) for s in sets] #Turn all the sets back into lists
    return merged #In summary: defines function that merges any lists (clusters) that
share a common element (by the logic that if A is near B and B is near C, then A is
"near" C)

mergeClusters = merge(allClusters)

clustMembers = []
for entry in mergeClusters:
    for value in entry:
        clustMembers.append(value) #Reduce list of cluster-lists down to a straight list (no
sublists)

for FID in inFIDs:
    if FID not in clustMembers:
        inIndex = inFIDs.index(FID)
        inCounterpart = nearFIDs[inIndex]

```

```
cluster4 = [FID,inCounterpart]
mergeClusters.append(cluster4) #If FID isn't in this list (i.e. it's not part of a cluster
>2 members), find that pair & include it in list of clusters
```

```
arcpy.MakeFeatureLayer_management(openSpace,"openSpaceLyr")
k = 0
for entry in mergeClusters: #Iterate over list of clusters
    k += 1 #Add 1 to k after each cluster, to keep track of filenames later
    for value in entry: #For each FID in cluster...
        whereClause = "FID = " + str(value) #Change the "where" clause
        arcpy.SelectLayerByAttribute_management("openSpaceLyr","ADD_TO_SELECTION",
        whereClause) #Build up a selection of all FIDs in the cluster
        arcpy.Dissolve_management("openSpaceLyr",os.path.join(dirString,"cluster_%s" %k))
#Dissolve them into 1 feature
        arcpy.SelectLayerByAttribute_management("openSpaceLyr","CLEAR_SELECTION")
#Clear selection to proceed to the next cluster in outermost "for" loop
```

```
k = 0
for entry in mergeClusters:
    k += 1
    for value in entry:
        whereClause = "FID = " + str(value)
        arcpy.SelectLayerByAttribute_management("openSpaceLyr","ADD_TO_SELECTION",
        whereClause)
arcpy.SelectLayerByAttribute_management("openSpaceLyr","SWITCH_SELECTION")
arcpy.CopyFeatures_management("openSpaceLyr",os.path.join(dirString,"unclustered.shp"
)) #Selects everything not in a cluster & exports that to a shapefile
```

```
polyList = []
parks = os.listdir(dirString) #List all files in "Underserved Areas" folder
parks2 = fnmatch.filter(parks,"*shp") #Filter them for just the .shp files
for park in parks2:
    polyList.append(dirString + "\\\" + park) #Append them to list with their full path
names
```

```
merged = arcpy.Merge_management(polyList,os.path.join(dirString,"GreenSpace.shp"))
#Create new shapefile of all green space w/ clusters now listed as 1 feature
```

```
arcpy.AddWarning("Allocation of service areas...")
servAreaRast = arcpy.sa.EucAllocation(merged,"","","","FID")
servAreas =
arcpy.RasterToPolygon_conversion(servAreaRast,"in_memory/servPoly","","VALUE")
servClip = arcpy.Clip_analysis(servAreas,shedBound,"in_memory/servClip")
```

```
arcpy.AddWarning("Removing any fields from previous runs...")
fields = []
```

```

fieldList = arcpy.ListFields(merged)
for field in fieldList:
    fields.append(field.name)
if 'Acreage' in fields:
    arcpy.DeleteField_management(merged,"Acreage")

fields2 = []
fieldList2 = arcpy.ListFields(parcels)
for field in fieldList2:
    fields2.append(field.name)
if 'ServPt' in fields2:
    arcpy.DeleteField_management(merged,"ServPt")

arcpy.AddWarning("Calculating green space acreages...")
arcpy.AddField_management(merged,"Acreage","FLOAT","","","","NON_NULLABLE","NO
N_REQUIRED","")
arcpy.CalculateField_management(merged,"Acreage","!shape.area@ACRES!","PYTHON_9.3
","")

arcpy.AddWarning("Calculation of green space per capita...")
blocksLayer = arcpy.MakeFeatureLayer_management(popBlocks,"blocksLyr","","","FID FID
VISIBLE NONE; Shape Shape VISIBLE NONE; TotalPop TotalPop VISIBLE RATIO")
servPops =
arcpy.Intersect_analysis([blocksLayer,servClip],"in_memory/servPops","ALL","","INPUT")
servPopsDiss =
arcpy.Dissolve_management(servPops,os.path.join(dirString,"servAreas.shp"),"grid_code",
TotalPop SUM")
arcpy.JoinField_management(servPopsDiss,"grid_code",merged,"FID","Acreage")

arcpy.AddField_management(servPopsDiss,"ParkPerCap","FLOAT","","","","NON_NULLA
BLE","NON_REQUIRED","")
arcpy.CalculateField_management(servPopsDiss,"ParkPerCap","!ACREAGE! /
!SUM_TOTALP!","PYTHON_9.3","")

arcpy.AddWarning("Awarding 1 point to parcels in an underserved area...")
arcpy.AddField_management(merged,"ServPt","FLOAT","","","","NON_NULLABLE","NON_
REQUIRED","")

areaList = []
rows = arcpy.SearchCursor(servPopsDiss","","","ParkPerCap","ParkPerCap A")
for row in rows:
    areaList.append(row.ParkPerCap)
del row, rows

def percentile(areaList,P):
    n = int(round(P * len(areaList) + 0.5))

```

```

if n > 1:
    return areaList[n-2]
else:
    return 0

cutoff = percentile(areaList,float(P))

arcpy.MakeFeatureLayer_management(servPopsDiss,"servLyr")
arcpy.MakeFeatureLayer_management(parcels,"parcelLyr")
selected =
arcpy.SelectLayerByAttribute_management("servLyr","NEW_SELECTION","ParkPerCap <=
" + str(cutoff))
selParcels =
arcpy.SelectLayerByLocation_management("parcelLyr","HAVE_THEIR_CENTER_IN",selecte
d) #Selects parcels with their centroid within underserved areas (those with
"ParksPerCap" <= user-specified cutoff value)

rows = arcpy.UpdateCursor(selParcels,"","","ServPt")
for row in rows:
    row.ServPt = 1
    rows.updateRow(row)
del row, rows

```

## Sediment erosion/deposition modeling with USPED

```
import os, sys, arcpy
from arcpy import env
from arcpy import sa

arcpy.CheckOutExtension("spatial")

DEM = sys.argv[1]
MExp = sys.argv[2]
KFact = sys.argv[3]
CFact = sys.argv[4]
RFact = sys.argv[5]
sumUnit = sys.argv[6]
parcels = sys.argv[7]
workspace = sys.argv[8]

arcpy.env.overwriteOutput = True
arcpy.env.workspace = workspace
arcpy.env.snapRaster = DEM
arcpy.env.extent = DEM
arcpy.env.cellSize = DEM
spatialRef = arcpy.Describe(DEM).spatialReference
arcpy.env.outputCoordinateSystem = spatialRef

description = arcpy.Describe(DEM)
cellSize = description.children[0].meanCellHeight

arcpy.AddWarning("Creating new directory...")
sedDir = arcpy.CreateFolder_management(workspace,"Erosion")
sedDirString = str(sedDir)

arcpy.AddWarning("Calculating flow accumulation, slope, and aspect grids...")
DEMDir = arcpy.sa.FlowDirection(DEM)
DEMFlow = arcpy.sa.FlowAccumulation(DEMDir)
DEMSlope = arcpy.sa.Slope(DEM,"DEGREE")
SlopeRad = DEMSlope * 0.01745 #Convert slope grid to radians
DEMAsspect = arcpy.sa.Aspect(DEM)
SinSlope = arcpy.sa.Sin(SlopeRad) #Sin of the slope, for use in  $A^m(\sin(b))^n$ 

#Calculation of  $A^m(\sin(b))^n$ 
arcpy.AddWarning("Calculating influence of incoming flow...")
LS = ((DEMFlow * cellSize / 22.13) ** MExp) * ((SinSlope / 0.0896) ** 1.3)

#qsy & qsx
arcpy.AddWarning("Calculating divergence of sediment transport capacity...")
Aspect2 = DEMAsspect * (-1) + 450
```

```

AspectRad = Aspect2 * 0.01745
CosAspect = arcpy.sa.Cos(AspectRad)
SinAspect = arcpy.sa.Sin(AspectRad)
qsx = LS * KFact * CFact * RFact * CosAspect
qsy = LS * KFact * CFact * RFact * SinAspect

```

### **#qsx\_dx & qsy\_dy**

```

arcpy.AddWarning("Calculating rate of change of sediment transport capacity...")
qsxSlope = arcpy.sa.Slope(qsx,"DEGREE")
qsySlope = arcpy.sa.Slope(qsy,"DEGREE")
qsxAspect = arcpy.sa.Aspect(qsx)
qsyAspect = arcpy.sa.Aspect(qsy)
qsxAspect2 = qsxAspect * (-1) + 450
qsyAspect2 = qsyAspect * (-1) + 450
qsxSlopeRad = qsxSlope * 0.01745
qsySlopeRad = qsySlope * 0.01745
qsxAspectRad = qsxAspect2 * 0.01745
qsyAspectRad = qsyAspect2 * 0.01745
CosAspectX = arcpy.sa.Cos(qsxAspectRad)
SinAspectY = arcpy.sa.Sin(qsyAspectRad)
TanSlopeX = arcpy.sa.Tan(qsxSlopeRad)
TanSlopeY = arcpy.sa.Tan(qsySlopeRad)
qsx_dx = CosAspectX * TanSlopeX
qsy_dy = SinAspectY * TanSlopeY

```

### **#E**

```

arcpy.AddWarning("Calculation of erosion/deposition index...")
E = qsx_dx + qsy_dy
E.save(os.path.join(sedDirString,"ErosDepos.img"))
E = os.path.join(sedDirString,"ErosDepos.img")

```

### **#Summarize by subwatershed**

```

arcpy.AddWarning("Summarizing by management unit...")
unitStats =
arcpy.sa.ZonalStatisticsAsTable(sumUnit,"SUBBASIN",E,"EIndex.dbf","DATA","MEAN")
EIndexTable = os.path.join(workspace,"EIndex.dbf")

```

```

fields = []
fieldList = arcpy.ListFields(sumUnit)
for field in fieldList:
    fields.append(field.name)
if u'MEAN' in fields:
    arcpy.DeleteField_management(sumUnit,"MEAN")

```

```

arcpy.JoinField_management(sumUnit,"SUBBASIN",EIndexTable,"SUBBASIN","MEAN")

```

```

arcpy.AddWarning("Awarding 1 point to parcels in high-erosion management unit...")
fields2 = []
fieldList2 = arcpy.ListFields(parcels)
for field in fieldList2:
    fields2.append(field.name)
if u'SedimPt' in fields:
    arcpy.DeleteField_management(parcels,"SedimPt")

arcpy.AddField_management(parcels,"SedimPt","FLOAT","","","","NON_NULLABLE","NO
N_REQUIRED","")

arcpy.MakeFeatureLayer_management(sumUnit,"sumUnitLyr")
arcpy.MakeFeatureLayer_management(parcels,"parcelLyr")
selected =
arcpy.SelectLayerByAttribute_management("sumUnitLyr","NEW_SELECTION","MEAN <= "
+ str(-1))
selParcels =
arcpy.SelectLayerByLocation_management("parcelLyr","HAVE_THEIR_CENTER_IN",selecte
d)

rows = arcpy.UpdateCursor(selParcels,"","SedimPt")
for row in rows:
    row.SedimPt = 1
    rows.updateRow(row)
del row, rows

```