**Gnarly Landscape Utilities:**

**Core Mapper**

**User Guide**

*Version 1.0—Updated October 2013*

Andrew J. Shirk1 and Brad H. McRae2

1The University of Washington Climate Impacts Group

2The Nature Conservancy

**Preferred Citation**

Shirk, A.J., and B.H. McRae. 2013. Gnarly Landscape Utilities: Core Mapper User Guide. The Nature Conservancy, Seattle WA. Available at: <http://www.circuitscape.org/linkagemapper>.

**Acknowledgements**

We thank the Washington Habitat Connectivity Working Group for feedback on these tools as they were being developed.

**Software Requirements**

The Gnarly Landscape Utilities require AcrGIS 10 with the Spatial Analyst extension.

**Licensing**

Copyright (C) 2010 Andrew Shirk and Brad McRae

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License (available at <http://www.gnu.org/licenses/>) for more details.

**Overview**

Core Mapper was developed by Andrew Shirk (University of Washington Climate Impacts Group) and Brad McRae (The Nature Conservancy) for the Washington Wildlife Habitat Connectivity Working Group. Core Mapper was developed with support from the [Great Northern Landscape Conservation Cooperative](http://www.nrmsc.usgs.gov/gnlcc).

Core Mapper is an ArcGIS (ESRI) script (written with Python 2.6) designed to identify core habitat areas from a habitat value raster, an optional landscape resistance raster, and several user-specified parameters. The intended purpose of Core Mapper is to provide a reproducible and flexible framework for identifying key areas of a landscape between which it may be important to evaluate habitat connectivity. The cores identified by this tool may be used directly for mapping linkages and other connectivity models. Note that mapped cores are discretely bounded areas on the landscape more suited for the core-corridor concept of modeling habitat connectivity, as opposed to more continuous models that do not require discrete cores (e.g. resistant kernel and centrality/gradient approaches).

**Installation**

Core Mapper is provided as part of Gnarly Landscape Utilities, an ArcGIS toolbox that can be added to any ArcGIS map document from the toolbox window. In addition to the toolbox, we provide example data in the ‘CoreMapperDemo’ folder, including an example resistance layer, habitat layer, and input parameter spreadsheet.

To install, unzip the toolbox package into a directory of your choosing (make sure you have read/write permission for the directory). Then add the Gnarly Landscape Utilities toolbox to your toolboxes following standard procedures found on the ArcGIS help pages.

Core Mapper requires the Python package openpyxl . We have included a compatible version in the ‘Toolboxes’ subfolder of the Gnarly Landscape Utilities install directory. If you use the included version, make sure you keep it in the ‘toolbox\scripts’ directory that it came in (that’s where the toolbox expects to find it if it isn’t in the site-packages folder). You can also install it (available [here](http://openpyxl.readthedocs.org/en/latest/)) in the site-packages folder of your ArcGIS Python directory.

**Overview of Steps in Core Mapper Process:**

1. For each pixel in the landscape, Core Mapper calculates the average habitat value within a circular moving window (user-specified and often sized to approximate the area of a home range for the species) centered on that pixel.
2. Next, Core Mapper uses a user-specified threshold minimum average habitat value to convert the output of step 1 into a binary raster of core/non-core habitat.
3. Sometimes, pixels of poor habitat are still defined as core habitat after step 2 because they are surrounded by high quality habitat that increases the nearby average habitat quality above the threshold in step 1. Step 3 eliminates those pixels of poor habitat by allowing the user to specify a ‘per-pixel’ minimum habitat value threshold. Any core habitat pixels below this threshold are converted to non-habitat.
4. Next, Core Mapper (optionally) can expand outwards in cost-weighted distance from core habitat pixels up to a user-specified movement distance across the resistance surface to potentially link proximate patches of core habitat into larger aggregates. This results in fewer but larger cores. By definition, if this step does not join habitat patches, it would require movements that exceed twice the user-specified cost-weighted distance. Unlinked cores after this step are therefore separated by longer-distance movements, which can be modeled using core-corridor approaches such as Linkage Mapper.
5. If step 4 was implemented, Core Mapper then optionally removes pixels from expanded cores with moving window averages that are below the user-specified minimum average habitat value specified in step 2. This shrinks cores back, but can leave some core area pairs connected that were disjunct before step 4.
6. Next, Core Mapper will remove cores that are below a user-specified minimum area (expressed in square map units). If cores were expanded outward (step 4 above), they may include pixels below the user-specified ‘per-pixel’ minimum value. The user has the option to exclude these pixels from area calculations.

**Core Mapper inputs**

When you run the Core Mapper tool in ArcGIS, you will be asked to navigate to one or more spreadsheets (in .xlsx format). Loading multiple spreadsheets offers a means to batch run many inputs. Within each spreadsheet, parameters are entered in a single row (starting with row 2). Entering parameters in multiple rows (row 2+, with no spaces between rows) is also a means to batch run multiple models, with each row run sequentially in order. Each spreadsheet specifies all the parameters Core Mapper needs to run, including:

*Habitat value raster* – an ESRI raster grid representation of habitat value (e.g., a grid created using the Resistance and Habitat Calculator tool). The grid values can be scaled to any range so long as the higher values represent higher habitat value (or higher landscape integrity/lower human modification value if using that approach). The inputs ‘minimum average habitat value of home range’ and ‘minimum habitat value per pixel’ below are based on the range of habitat values in this raster, so please choose these inputs appropriately. This raster should be clipped to an extent encompassing your study area. Very large raster grids will increase processing time.

*Resistance raster* – an ESRI raster grid representation of landscape resistance (can also be created using the Resistance and Habitat Calculator tool). The grid values should have a minimum value of one. There are no limits to the maximum value. Resistance values act as a multiplier, so a resistance of two for a 100m cell, for example, would equate to 200m in cost-weighted distance. Thus, higher resistance values reduce movement proportionally. This raster should be clipped to the same extent as the habitat value raster and also be in the same coordinate system/projection. If you’re not expanding core habitat (Step 4) you can leave this blank.

*Output directory* – please specify a directory where the output files (including intermediate files in the ‘temp’ folder) will be written.

*Output filename* – please specify a name which will be used to name the outputs.

*Moving window radius* – this value represents the search radius for average habitat value calculations, often corresponding to the home range size of the species of interest (in which case it would be calculated as ). This value should be expressed in the same map units as the rasters above (e.g. a radius of 2 km should be input as 2000 if your map units are meters).

*Minimum average habitat value* – this value represents the minimum average habitat value within a circular moving window to be considered core habitat. If the average habitat value of the window centered on a particular grid cell is less than this minimum, that grid cell will not be included within the final cores. This minimum value should be based on the range of habitat values in the habitat raster.

*‘Per pixel’ minimum habitat value* – this value is used to reclassify the habitat raster into a binary core habitat map. Only grid cells meeting the ‘minimum average home range habitat value’ requirement above will be evaluated. For those cells meeting this criteria, any pixel with a habitat value greater than the per-pixel minimum value will be considered core habitat.

*Minimum core area* – cores that meet all other requirements but are below this minimum area will be excluded from the final output shapefile. Minimum core areas should be expressed in map units squared. For example, a minimum size of 25 km2 expressed in map units of meters would be 25,000,000 (1km = 1000000 m2).

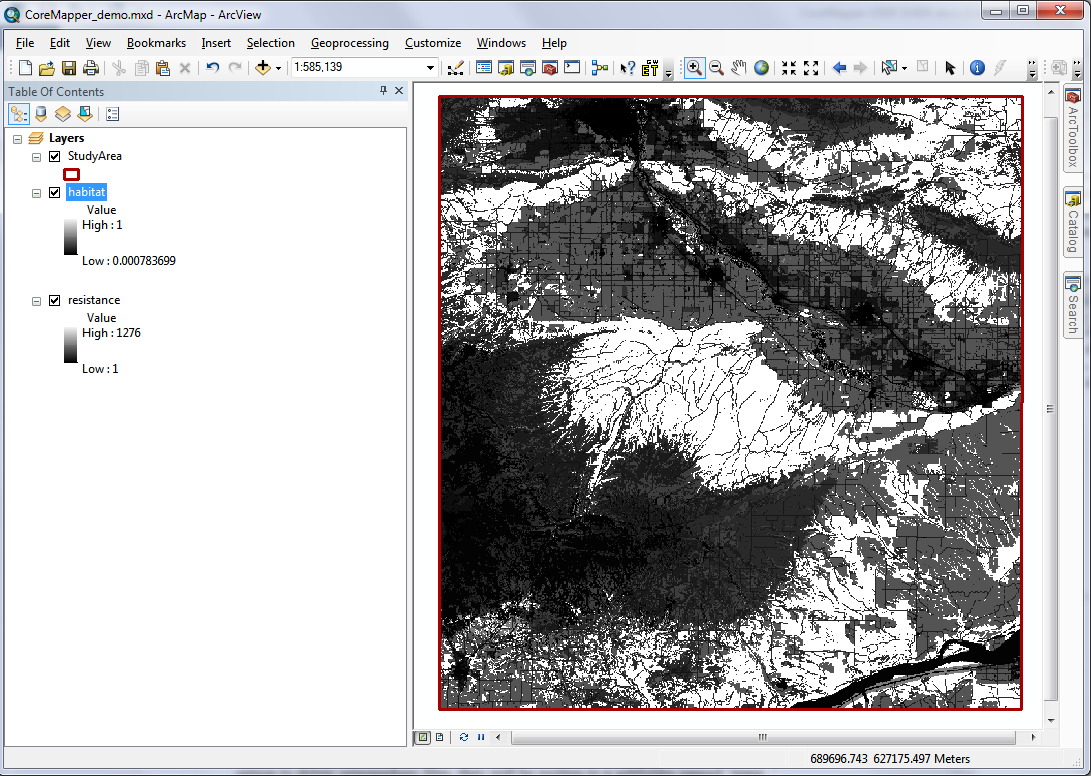
**Core Mapper outputs**

Core Mapper will write a shapefile (with its name derived from the first column in the input spreadsheet) to the ‘cores’ subfolder created in the user-specified output folder (also specified in the input spreadsheet), and a log file in the ‘log’ subfolder of the output folder. This shapefile has attributes including core\_ID (a unique identifier for each core), number of cells, area (in square map units), and minimum, maximum, range, mean, standard deviation, and sum of the values from the habitat model within the core.

In addition, a series of temporary files will be written to a subfolder named ‘temp\_’ followed by the output filename you specified in the input spreadsheet. Several of these temporary outputs are described in the tutorial below. They can be useful to review so you can understand how your parameterization of the core models affects the outcome.

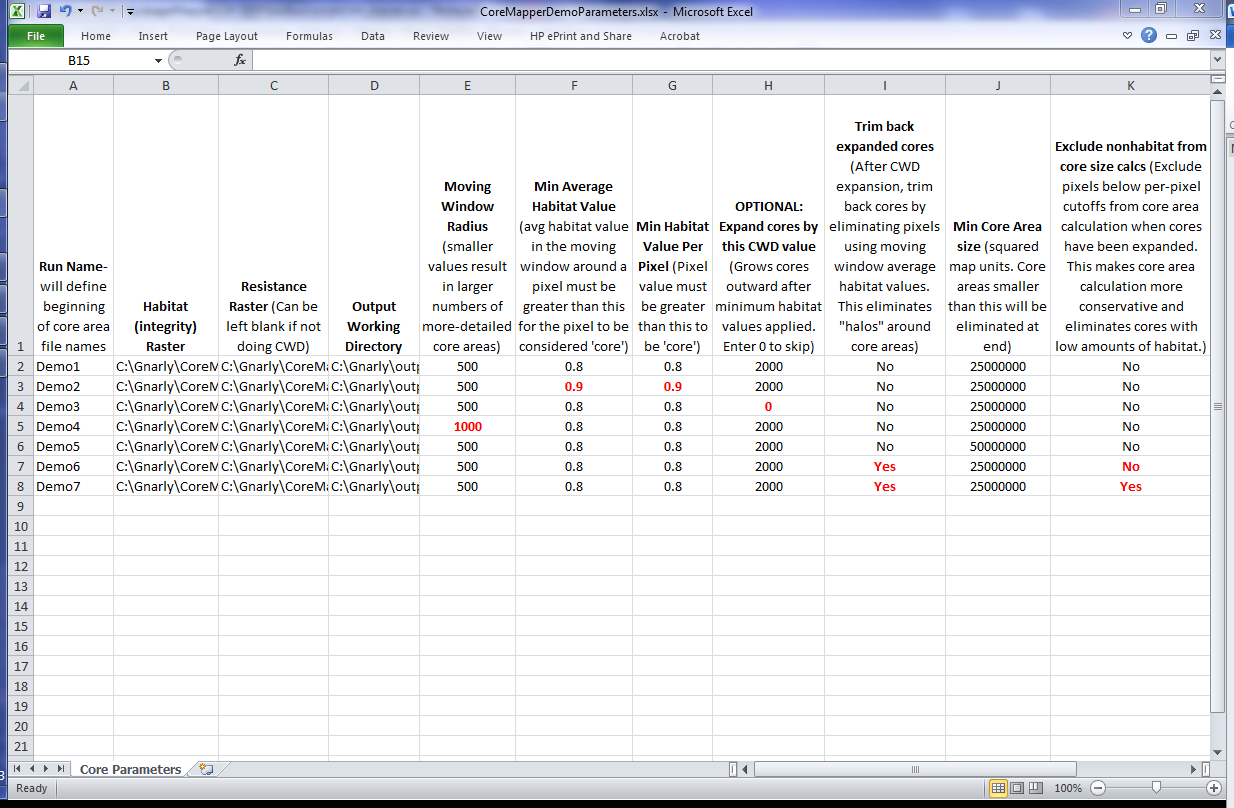
**Tutorial**

To start this tutorial open the CoreMapper\_demo.mxd ArcMap document in the ‘CoreMapperDemo’ folder. This map has the files you’ll need and the Gnarly toolbox already loaded, including a habitat model (raster), a resistance model (raster), and a shapefile defining the boundary of the study area.



**Figure 1. Habitat raster.**

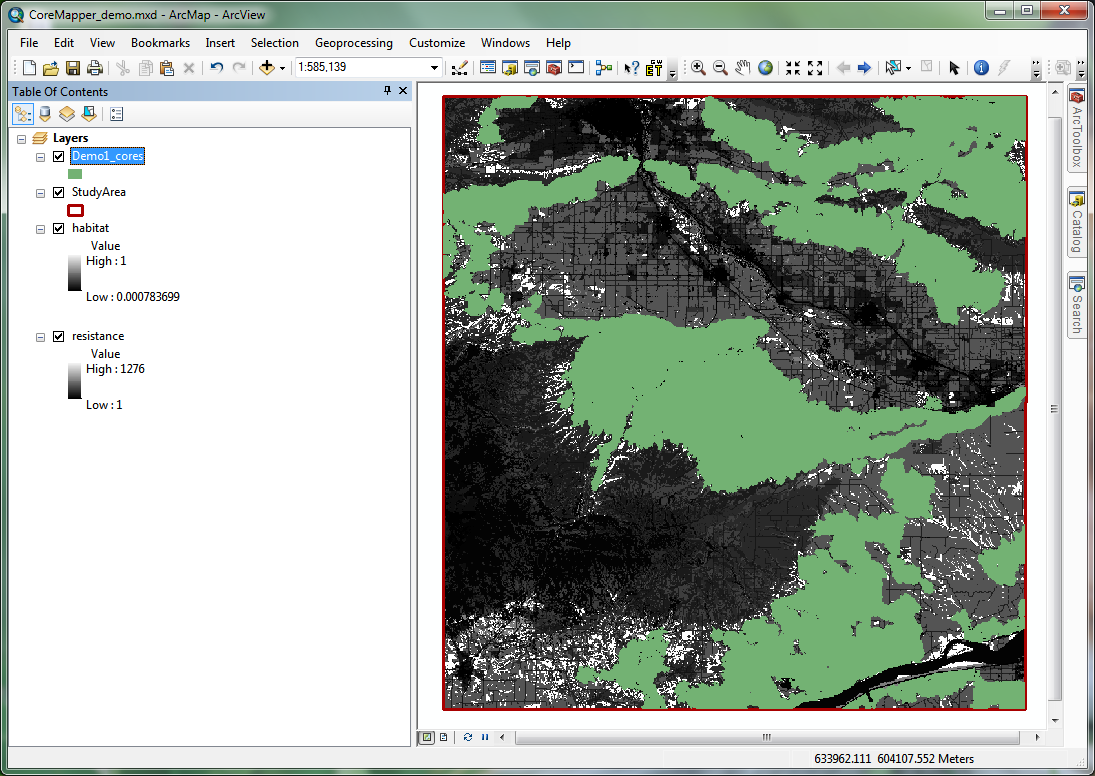
Now open the CoreMapperDemoParameters.xlsx spreadsheet in the ‘CoreMapperDemo’ folder. Here you’ll see all the parameters that need to be set for Core Mapper to run. Get familiar with the definitions of these inputs before we explore how they affect the core outputs. Note that each line in the input spreadsheet parameterizes a new set of core outputs. So if you want to explore a range of options, you simply add a line for each in the input file and the tool will batch run them all sequentially. For now, do not change any parameters in the input spreadsheet, except the directory paths to the habitat raster, resistance raster, and desired output folder, if different from what is already entered.



**Figure 2. Demo Excel spreadsheet.**

Now open up the Core Mapper tool in the Gnarly toolbox preloaded into your demo map document. In the tool dialog, simply browse to the CoreMapperDemoParameters.xlsx spreadsheet, and click OK. Note that you can also perform batch runs via this menu by loading multiple spreadsheets. You can follow the progress of the Core Mapper run by the on-screen dialogue.

When the run is complete, add the Demo1\_cores.shp shapefile to the map, which can be found in the ‘cores’ subfolder of the output folder specified in the parameters spreadsheet (the original path was ‘C:\ Gnarly\output’). These are the cores based on the parameters from the first row of the spreadsheet.

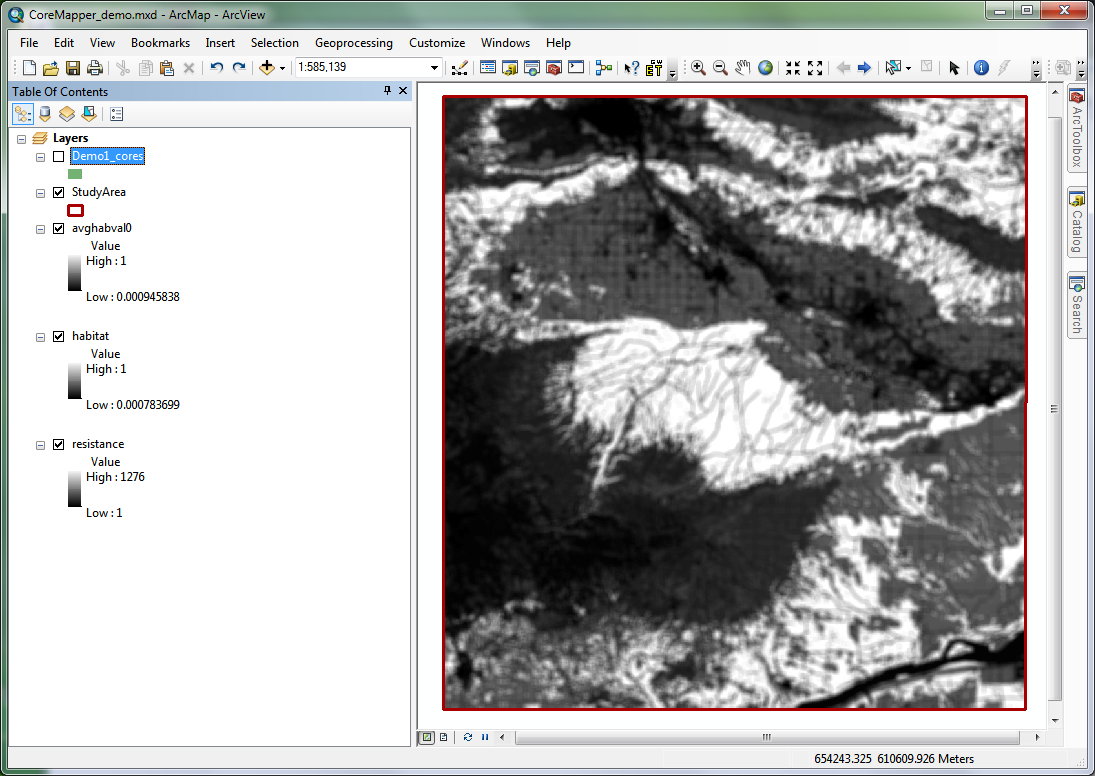


**Figure 3. Results from Demo1 run.**

Let’s explore how the input parameters were used to derive this core model. This process occurs over four major steps:

**STEP 1:**

The first step in the process is to calculate the average habitat value within a user-specified circular moving window radius (in this run, it was 500m). Add the ‘avghabval’ raster in the ‘temp\_Demo1’ folder of the output directory to your map to see this result.



**Figure 4. Average habitat values in 500-m moving window.**

Compare this result with the habitat model, and notice how it’s been smoothed by the moving window. The larger the moving window radius, the greater the smoothing effect.

**STEP 2:**

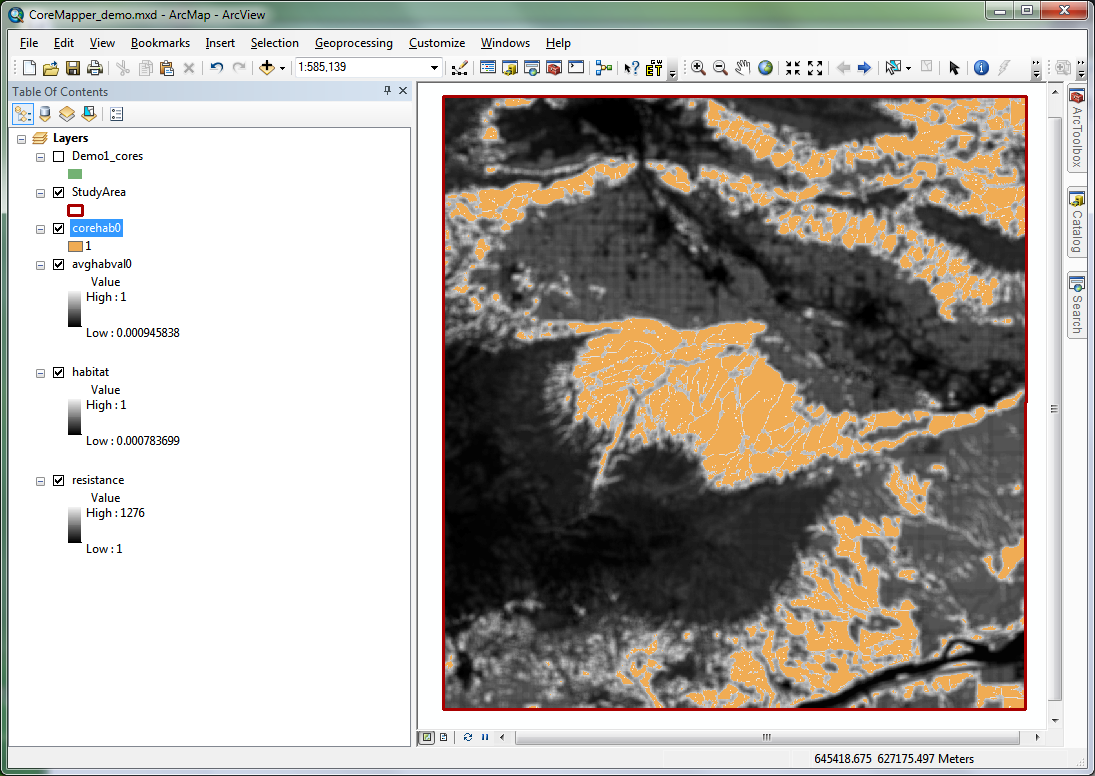
The next step in the process involves putting a threshold (minimum average habitat value) on the moving window result from step 1, such that if a cell has an average habitat value within 500m that is below the threshold, it is made null (i.e. takes on a nodata value) and if the moving window average is above the threshold, it takes on a value of 1.

We suggest defining the moving window radius based on a circle with an area approximating the home range for the species being studied. Defining the moving window this way means the result of STEP 2 can be interpreted as portions of the landscape where the abundance and quality of habitat within an animal’s home range habitat is sufficient to meet its needs. The higher the threshold, the smaller the area that is ‘in play’ to become cores.

**STEP 3:**

Even though some pixels may have a high average habitat value within the moving window, they may still be of low suitability themselves. These low-suitability pixels can be removed in this step by setting a threshold (minimum habitat value per-pixel), below which they will be reclassified as null.

STEP 3 removes areas of low per-pixel suitability. The result of STEP 2 and STEP 3 can be seen in the ‘corehab’ raster in the ‘temp’ output directory (orange in the figure below).



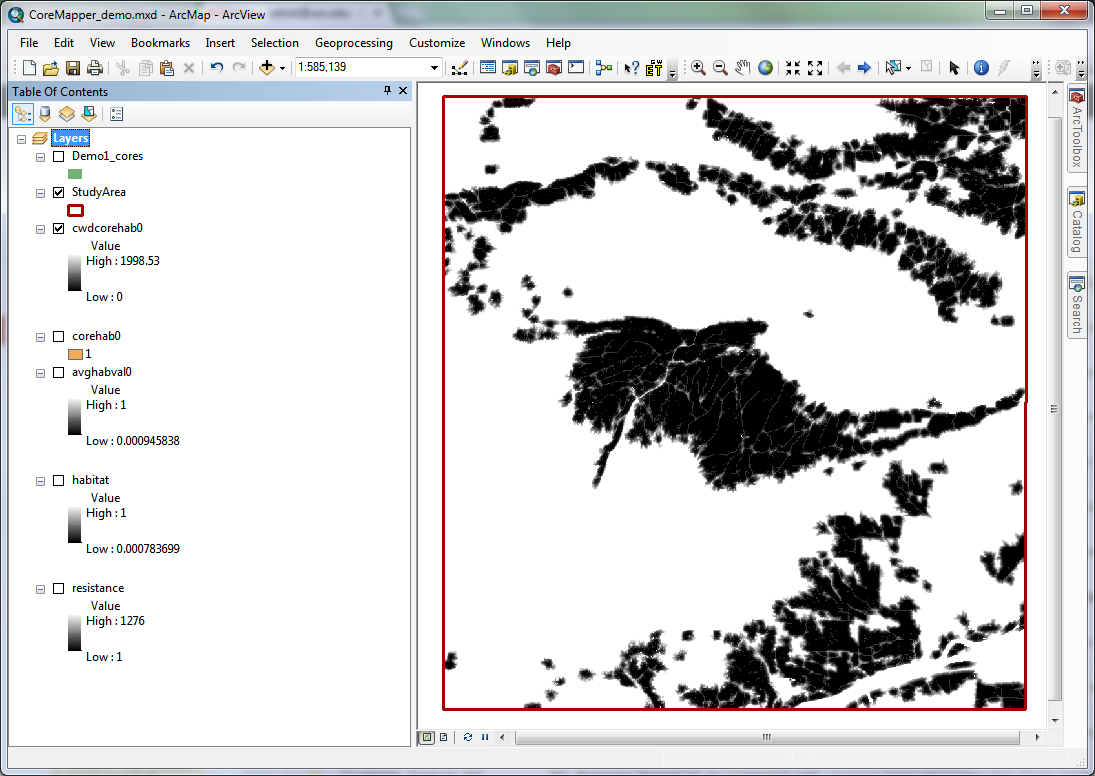
**Figure 5. Binary habitat raster following step 3.**

Compare the ‘corehab’ layer to the original habitat layer (not the moving window average). Notice that there are quite a few high suitability habitat pixels that aren’t included in these initial cores because they are surrounded by too much low-suitability habitat.

**STEP 4:**

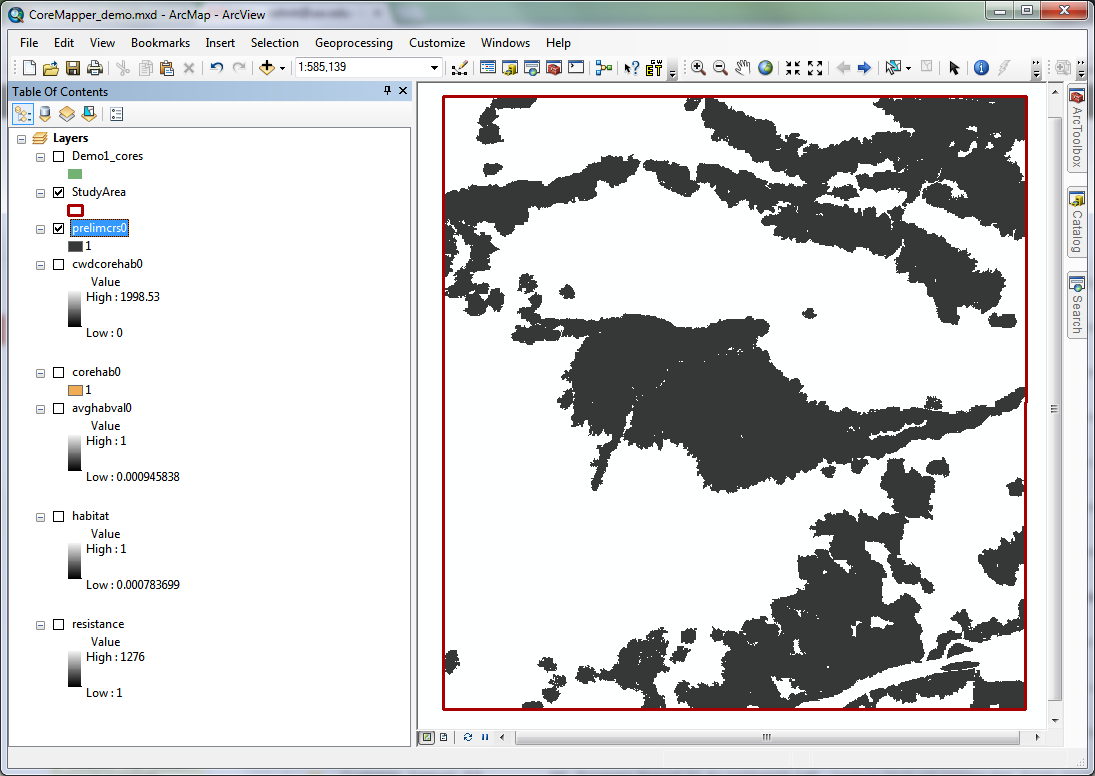
The output from STEP 2 is now starting to look like a core habitat model, but notice that there are many cores in close proximity with low resistance between them (based on the resistance model included in your demo data folder). In this step, we define a short range movement distance based on the species under study. This distance distance is designed to capture the ability of species to move between adjacent patches of habitat as part of its home-range movements. Any movements beyond this threshold are considered longer distance dispersal movements, and may be modeled via corridor tools like Linkage Mapper.

Core Mapper will expand the edges of the cores from STEP 2 outwards up to the distance specified, using the resistance model to weight the distances (i.e. a resistance of 2 uses up the allotted distance twice as fast as a resistance of 1). This ‘cost-weighted distance’ model may be found in the ‘temp’ output folder as the ‘cwdcorehab’ raster.



**Figure 6. CWD-expanded habitat raster.**

Another part of this step converts the above cost-weighted distance model to a new raster (‘prelimcrs’ in the ‘temp’ output directory) where all pixels included in preliminary cores have a value of 1.



**Figure 7. New binary habitat raster based on CWD-expanded raster.** Notice how the preliminary cores after STEP 3 have merged together into fewer and slightly larger cores, rather than many small cores separated by short distances.

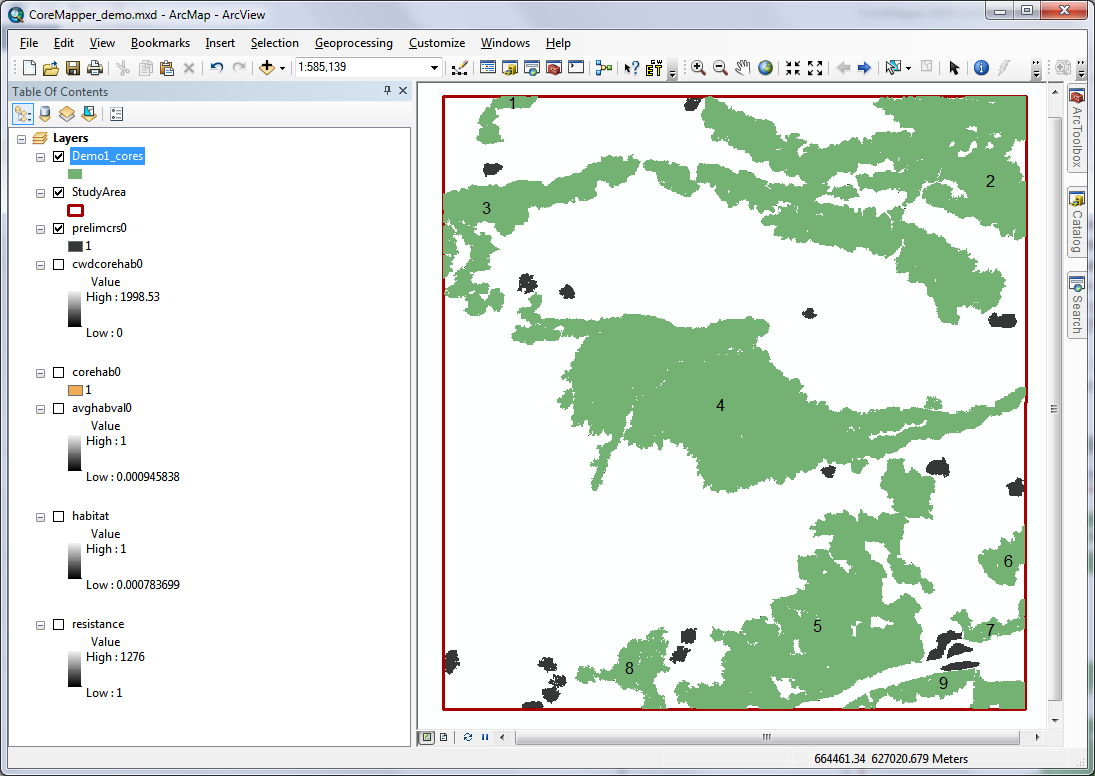
**STEP 5:**

We’ll skip step 5 for now, since cores were not trimmed back after CWD expansion. We’ll return to this in the demo6 and demo7 runs.

**STEP 6:**

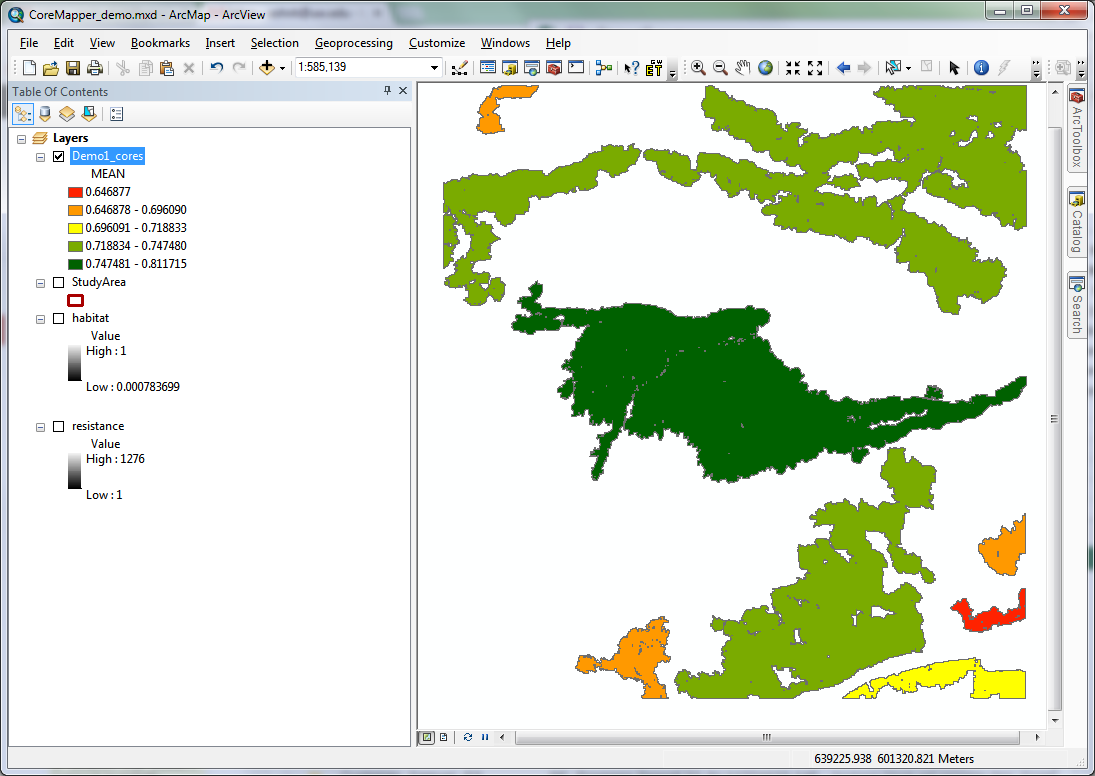
The final step involves removing patches of core habitat that are too small to be relevant for your study. This threshold area could be based on the minimum area to support a viable subpopulation (if connected to other patches via dispersal). Alternatively, you could choose a minimum size threshold to reach a desired number of cores. In this run of Core Mapper, we’re removing all cores with an area less than 25 km2 (remember in the parameter table to specify area units in the same units of your raster models – here we used 25 million square meters).

This minimum area threshold produces the final core model in the ‘cores’ output directory. Notice in the image below the small dark patches that did not meet this threshold and were removed.



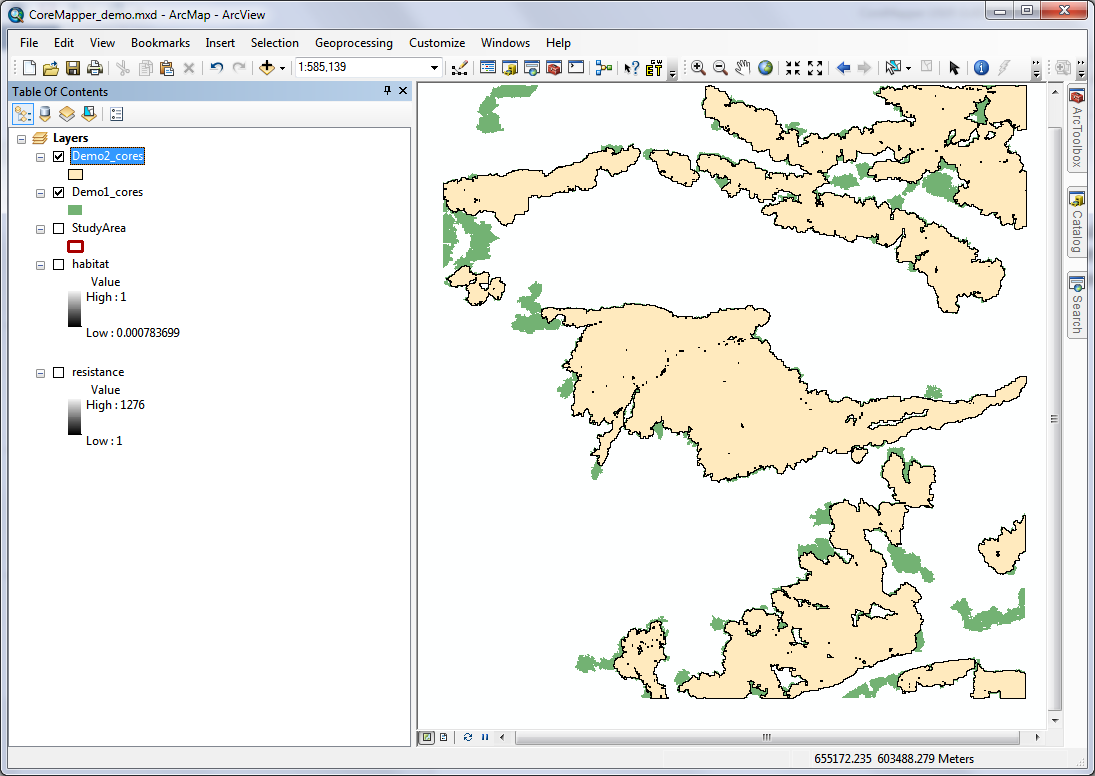
**Figure 8. Demo1 results on top of preliminary cores.**

Each of the 9 cores in the final output model has a unique identifier (core\_ID) in the attribute table. Notice the other attributes, including number of cells, area (in square map units), and minimum/ maximum/ range/ mean/ standard deviation/ sum of the values from the habitat model within the core. These can be useful for comparing cores based on their size or summary statistics of habitat suitability (the figure below shows the mean habitat value per core).



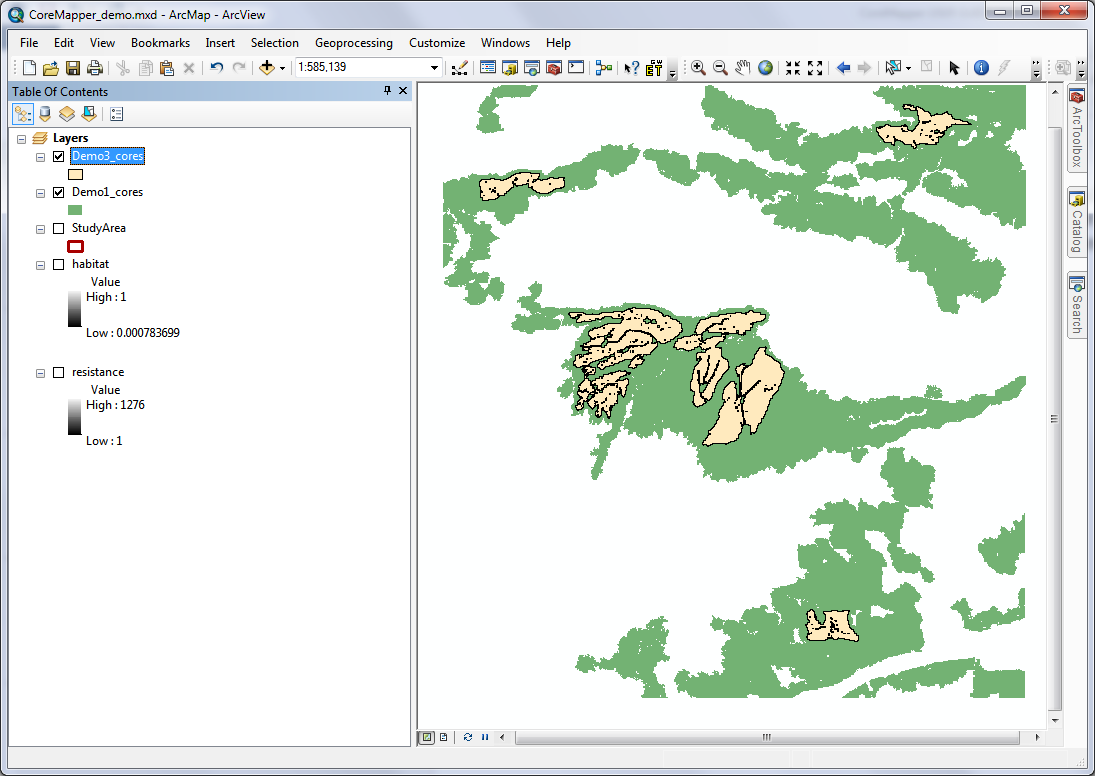
**Figure 9. Mean habitat value per core in Demo1 run.**

Now explore the other four model runs in the output folder to this first run. In each case, one of the input parameters was changed, resulting in rather profound differences in the cores produced. Demo2, for example, produced somewhat more restricted cores, because the thresholds for minimum habitat value were set higher (0.9).



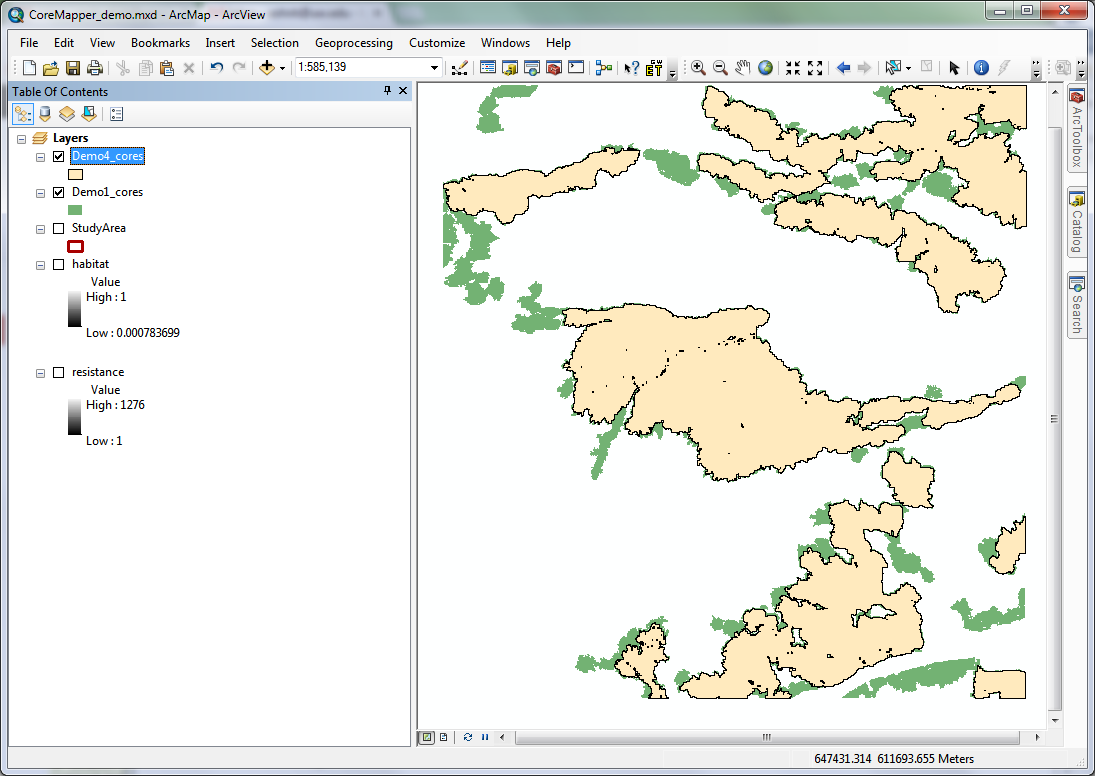
**Figure 10. Demo2 results.**

Demo3 specified a short-range movement distance of zero, which prevented nearby cores from growing together. Most of these smaller cores did not meet the minimum size requirement, so in the final model, only the largest initial (unmerged) patches were retained.



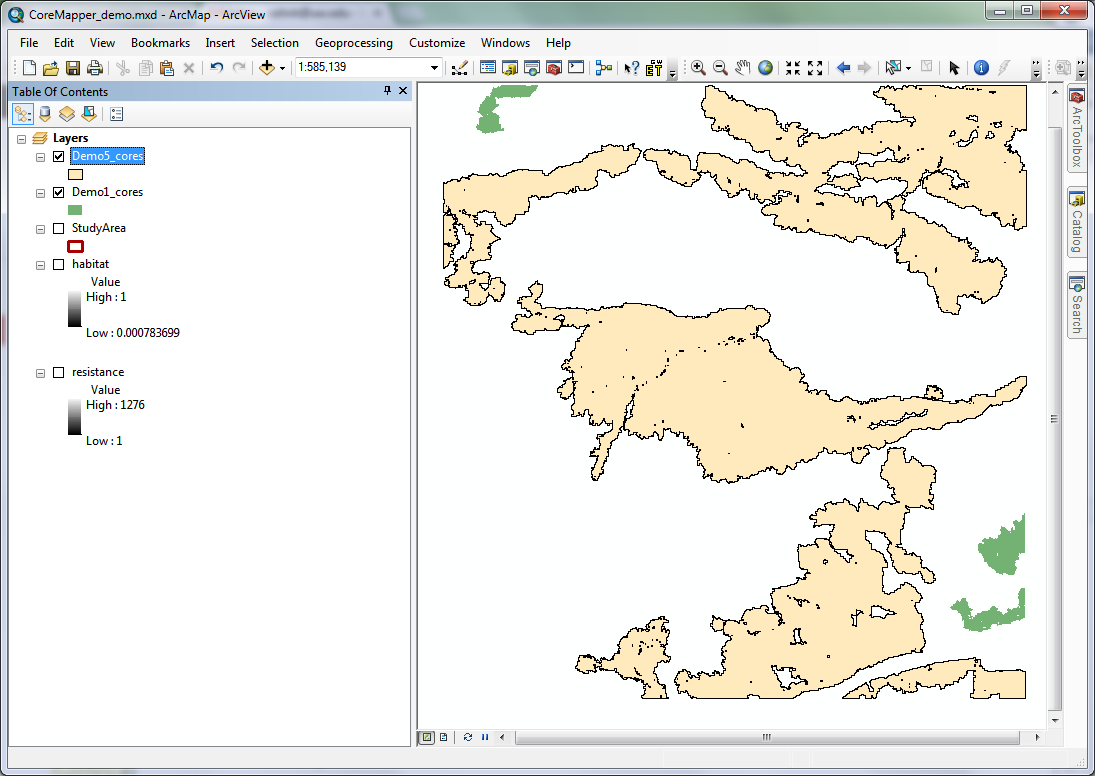
**Figure 11. Demo3 results.**

Demo4 increased the moving window radius. This can make it more difficult for extremities of core habitat areas to meet the minimum average habitat value requirement. The result is a somewhat smoother and more restricted set of cores.



**Figure 12. Demo4 results.**

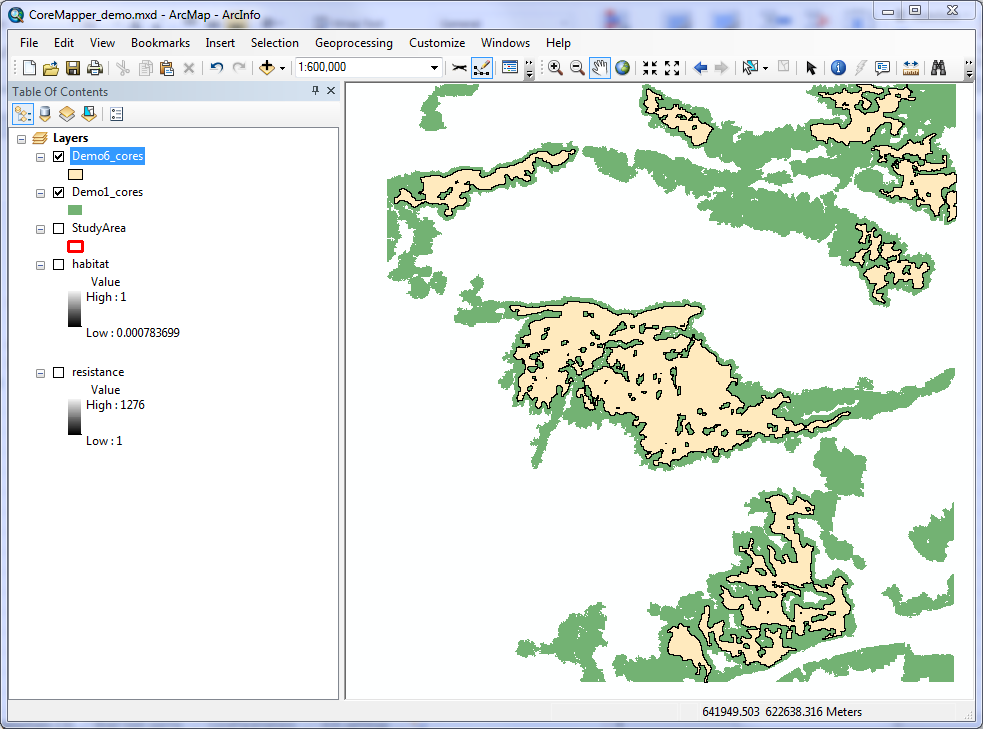
The demo5 cores had a larger minimum size requirement than the demo1 cores.



**Figure 13. Demo5 results.**

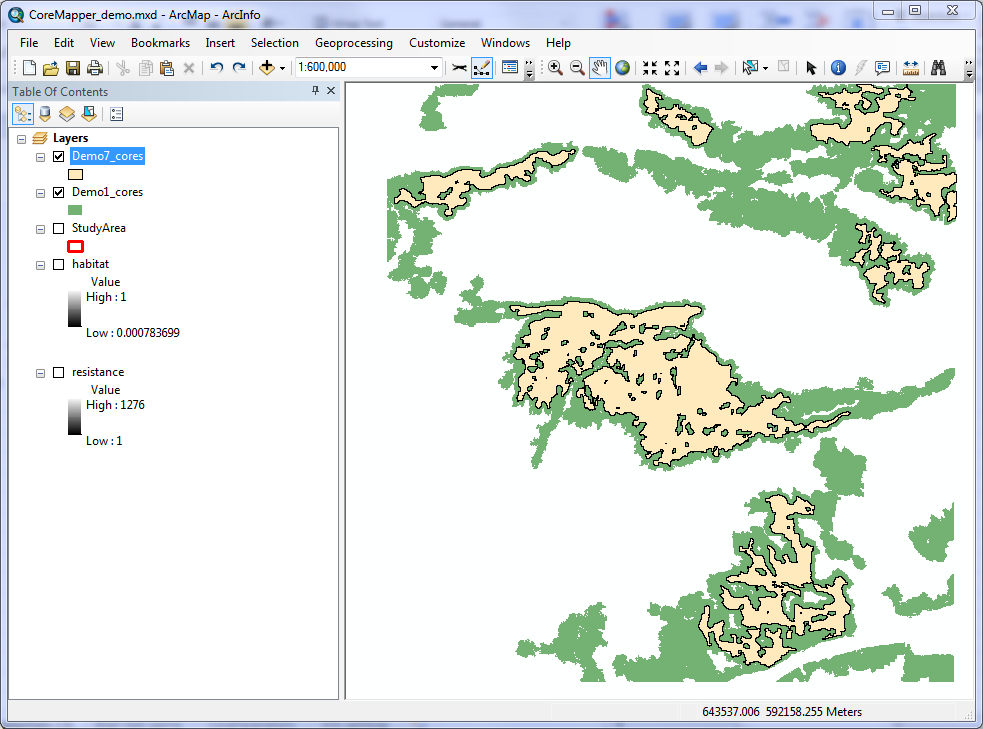
This larger minimum size produced identical cores except the smallest three cores were moved because they were too small.

The demo6 example invoked **Step 5,** in which cores were trimmed back after CWD expansion to eliminate any pixels where the moving window average habitat value fell below the Min Average Habitat Value parameter. Pixels that fall below the per-pixel minimum habitat value are still retained, so some cores still grow together.



**Figure 14. Demo6 results.**

The demo7 example also invoked Step 5,but the ‘exclude nonhabitat from core size calcs’ option was enabled. This means that pixels that fell below the per-pixel minimum habitat value were not included in area counts (note that these are still retained in cores, however). This resulted in one core area being eliminated because it didn’t have enough high quality pixels.



**Figure 15. Demo7 results.**

**Helpful Hints**

***Experimenting with settings***

We strongly recommend that users try multiple combinations of parameters to fully understand how each step works and to tune the model to their satisfaction.

***Saving and re-loading run settings***

ArcMap automatically saves settings from previous runs, and you can re-load them using the Results tab. See ArcMap help for instructions. You can pick up a failed run (or re-run starting from any step using different parameter choices) by un-checking the steps prior to the one you wish to start at. No need to delete output files; they will be automatically overwritten.

## Support

Please email Brad McRae ([mcrae@circuitscape.org](mailto:mcrae@circuitscape.org)) if you have trouble. Bug reports and suggestions for improvement are especially welcome.

**Further Reading**

Washington Wildlife Habitat Connectivity Working Group (WHCWG). 2010. *Washington Connected Landscapes Project: Statewide Analysis.* Washington Departments of Fish and Wildlife, and Transportation, Olympia, WA. Available at: <http://waconnected.org/statewide-analysis/>.

Washington Wildlife Habitat Connectivity Working Group (WHCWG). 2012. *Washington Connected Landscapes Project: Analysis of the Columbia Plateau Ecoregion.* Washington Departments of Fish and Wildlife, and Transportation, Olympia, WA. Available at: <http://waconnected.org/columbia-plateau-ecoregion/>

You can also find similar tools and modeling guidance on the [Corridor Design](http://corridordesign.org/) website.