Linkage Mapper User Guide

Version 0.6 BETA—Updated Sept 30 2011

Brad McRae and Darren Kavanagh
The Nature Conservancy

Acknowledgements
Brian Cosentino made substantial coding contributions to early versions of this toolkit. Viral Shah contributed code from his thesis for network analyses. Jeff Jenness supported this work with updates to the Conefor Inputs tool. Thanks to Andrew Gilmer and Theresa Nogeire for helpful comments on this user guide. Thanks also to the Washington Habitat Connectivity Working Group for feedback on these tools as they were being developed.

Software Requirements and Licensing

Linkage Mapper requires AcrGIS 9.3 with Spatial Analyst, Python 2.5 and NumPy (testing on ArcGIS 10 and Python 2.6 has been successful so far). This toolbox is provided free of charge under a GNU General Public License. More details can be found on the Linkage Mapper website, where our code is hosted: http://code.google.com/p/linkage-mapper/

Preferred Citation

McRae, B.H. and D.M. Kavanagh. 2011. Linkage Mapper Connectivity Analysis Software. The Nature Conservancy, Seattle WA. Available at: http://www.circuitscape.org/linkagemapper.

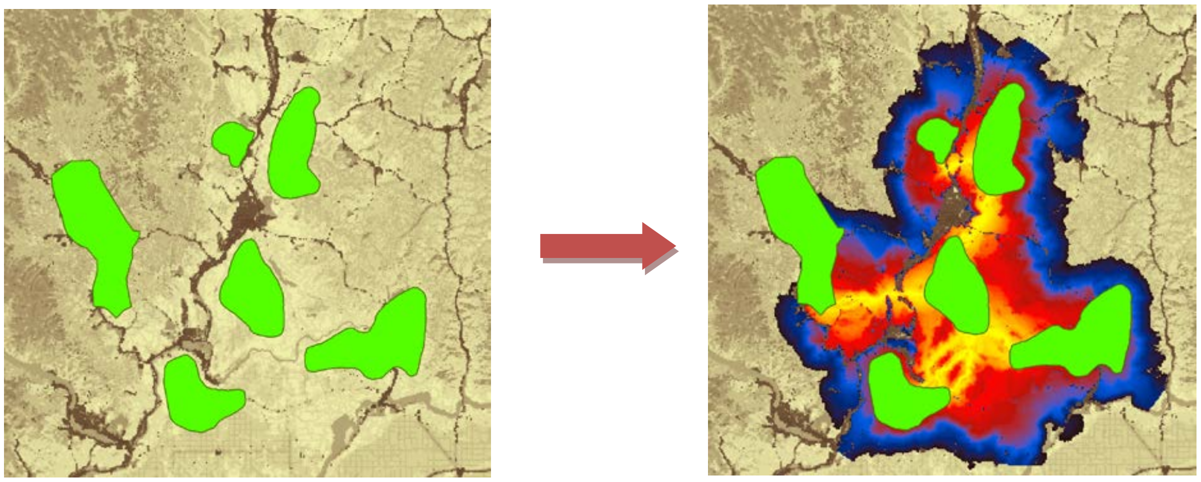


Table of Contents

1. Introduction

Linkage Mapper is a GIS tool designed to support regional wildlife habitat connectivity analyses. It consists of several Python scripts, packaged as an ArcGIS toolbox, that automate mapping of wildlife habitat corridors. We developed these scripts to support the 2010 Washington Wildlife Habitat Connectivity Working Group (WHCWG) statewide connectivity analysis, and are making them public for use in other wildlife connectivity assessments.

Linkage Mapper uses vector maps of core habitat areas and raster maps of resistance to movement to identify and map least-cost linkages between core areas. Each cell in a resistance map is attributed with a value reflecting the energetic cost, difficulty, or mortality risk of moving across that cell. Resistance values are typically determined by cell characteristics, such as land cover or housing density, combined with species-specific landscape resistance models. As animals move away from specific core areas, cost-weighted distance analyses produce maps of total movement resistance accumulated.

The scripts use ArcGIS and Python functions to identify adjacent (neighboring) core areas and create maps of least-cost corridors between them. The scripts then normalize and mosaic the individual corridor maps to create a single composite corridor map. The result shows the relative value of each grid cell in providing connectivity between core areas, allowing users to identify which routes encounter more or fewer features that facilitate or impede movement between core areas.

More details on the models and algorithms implemented by the Linkage Mapper can be found in Chapter 2 and Appendix D of WHCWG (2010).

Linkage Mapper is in active development. Please join the Linkage Mapper User Group for updates.

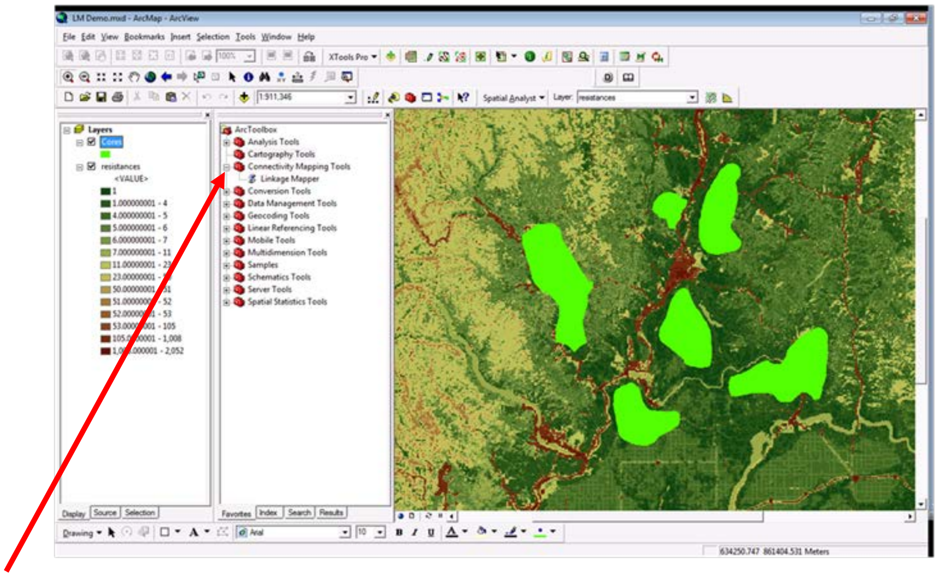
2. Installation

1) Install Linkage Mapper Beta
Download **Linkage Mapper from** http://code.google.com/p/linkage-mapper/

Open the linkagemapper.zip archive and place the *toolbox* directory with all subdirectories on your computer. Place the *LM_demo* directory in a folder that has no spaces or special characters in its path.

Open the demo project in ArcMap by double-clicking on *LM _Demo.mxd* in the *LM_demo* directory. Once ArcMap is active, go to Window>>ArcToolbox. Add the Linkage Mapper toolbox from the *toolbox* directory you just saved. To do this:

1. Right-click the ArcToolbox folder at the top of the window and select Add Toolbox.
2. Click the “Look in” dropdown arrow and navigate to the *toolbox* directory.
3. Click (don’t double-click) the Linkage Mapper toolbox.
4. Click open. The toolbox will be added to the ArcToolbox window.
5. Make sure the Spatial Analyst extension is activated by clicking:
    - Tools>>Extensions>>Spatial Analyst



Once you've added the Linkage Mapper toolbox, click the plus sign to show the contents of the folder.

2) Install Conefor Inputs Tool for ArcGIS
Download Conefor inputs from: http://jennessent.com/arcgis/conefor_inputs.htm

Once you've downloaded the tool, follow the instructions in the Conefor Inputs user guide. From there, you'll need to activate the Conefor toolbar in ArcMap: View>>Toolbars>>Conefor. The toolbar should look like this:

3) Make sure you have the required GIS and Python installations
Linkage mapper requires **ArcGIS 9.3 or 10 with Spatial Analyst.** You will also need **Python 2.5** or **2.6** and **NumPy** (Numerical Python), which are automatically installed by ArcGIS. If you do not have these, they can be found on the ArcGIS installation disk or downloaded if necessary.

4) Verify your installation
You can test the code by running the tutorial at the end of this document.

3. Using Linkage Mapper

3.1 Input data requirements

Inputs to Linkage Mapper include 1) a core area polygon GIS file, 2) a resistance raster GIS file, and 3) a text file specifying Euclidean (straight-line, edge-to-edge) distances between core area polygons. The core area polygon GIS file can be either an ESRI shapefile or ESRI geodatabase feature class. The file must have an attribute specifying core area IDs consisting of positive integers < 9999 that identify unique core areas. The resistance raster GIS file should include resistances represented by positive numbers (integers or floating point) only. We recommend resistances scaled so that values of 1 represent ideal habitat (WHCWG 2010) and increase to at least 100 for barriers (Beier et al. 2011).

The text file with Euclidean distances between cores is typically generated by the Conefor Inputs tool using the core area file as input (see next section).

3.2 Prepping your data and work spaces

Before you start, you'll need to set up your project directory and create a Euclidean distances file using your core area polygon data and the Conefor Inputs Tool.

1) Create a project directory

This is where intermediate and output files will be written. This should be a shallow and short directory (something like C:\ANBO), ideally on a local drive to speed processing. The name of the directory will also be used as a prefix for final output files (e.g. ANBO_LCPs). There should be *no spaces or special characters* anywhere in the directory path.

Note: running Linkage Mapper with input data or project directories on a shared drive can slow things down drastically. It's best to use a local drive if you can.

You should create unique project directories for each project/scenario you run, to avoid confusing intermediate data from different analyses.

Several subdirectories will be created in your project directory, including an *output* directory with final products. Data are passed between computational steps in the *datapass* directory. The scripts also create vector line maps with corridor statistics (stored in the *log* directory and the *link_maps* geodatabase in the *output* directory), as well as raster corridor maps (stored in the *corridors* geodatabase).

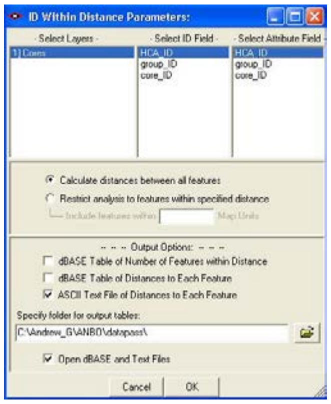
2) Calculate Euclidean (straight-line) distances between core areas using the Conefor Inputs Tool

The Conefor Inputs Tool measures minimum Euclidean distances between core area polygons. We use this tool solely to generate a text table of core area pairs and distances between them. If you have another way to provide a table of distances in the same format (see the demo data provided in the tutorial), you may do that instead.

Conefor Inputs gives you the option to analyze either all core area pairs or only those within a specified distance. Restricting analyses to a specified distance is particularly useful when working with large numbers of core areas for a specific species; if a species is only known to disperse *X* map units, and you want to only map linkages that are less than *X* in length, then using this cutoff can save calculation time if you have hundreds or thousands of core areas.

Don't forget: if you modify your core polygon layer, be sure to re-run Conefor Inputs with the new data.

Start by clicking on the left toolbar button


ID Within Distance Parameters:

- Select Layers -: 1) Cores
- Select ID Field -: HCA_ID, group_ID, core_ID
- Select Attribute Field -: HCA_ID, group_ID, core_ID

(● Calculate distances between all features) ○ Restrict analysis to features within specified distance
Include features within [ ] Map Units

-- -- -- Output Options: -- -- --
☐ dBASE Table of Number of Features within Distance
☐ dBASE Table of Distances to Each Feature
☑ ASCII Text File of Distances to Each Feature

Specify folder for output tables:
C:\Andrew_G\ANBO\datapass\

☑ Open dBASE and Text Files

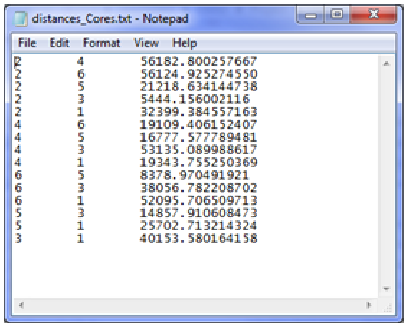
Cancel OK

This window should appear.

- Select your core area polygon file in the layers column. Then, select a unique core area ID field in the **ID** and **Attribute** field columns.
- Choose whether to calculate distances between all features or restrict your analyses to features within a specified distance (see above).

Note: If you restrict analyses to features within a specified distance, be sure to enter distances using the same units of measurement that are used in your GIS files.

- Have Conefor Inputs create an output file in text format by selecting **ASCII**.
- Select a location to save the output file. You'll need to browse to this file again later.
- Click OK (Conefor may give you an error, but you can ignore it as long as you get a distance table in your output folder).



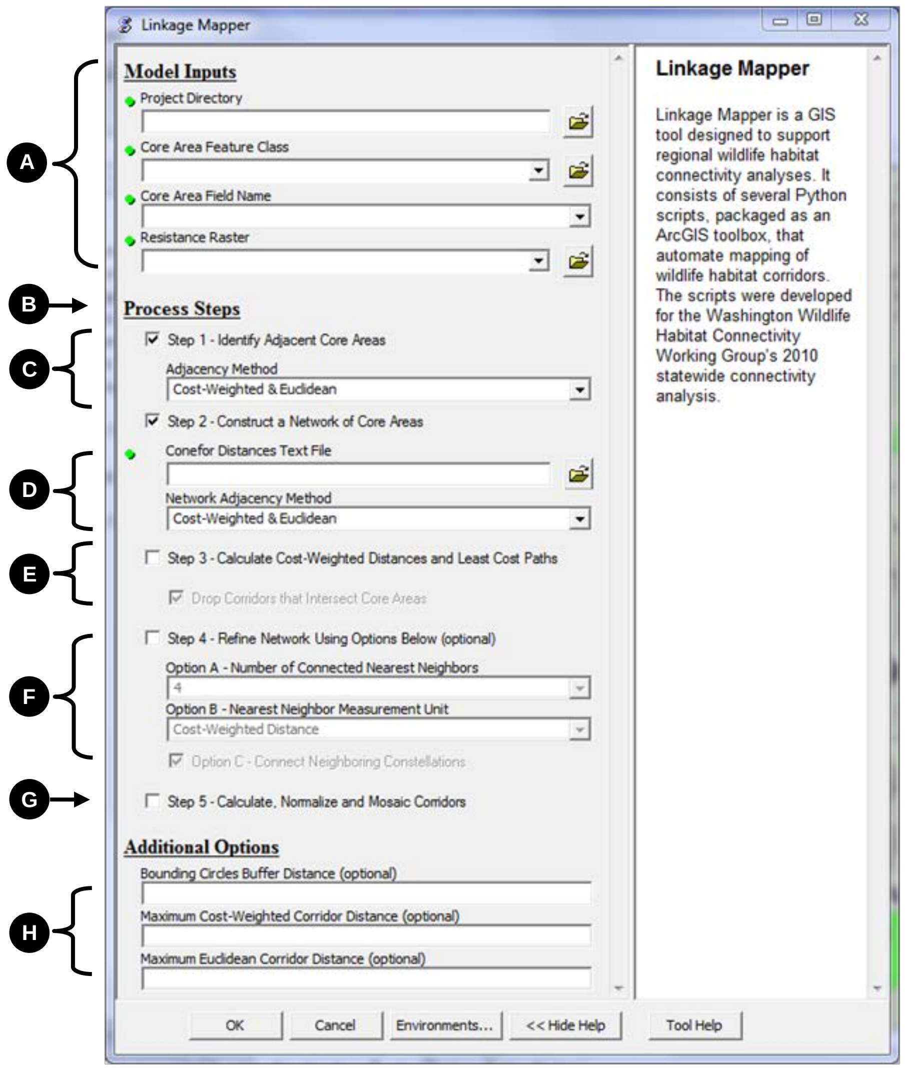
```
2       4        56182.800257667
2       6        56124.925274550
2       5        21218.634144738
2       3        5444.156002116
2       1        32399.384557163
4       6        19109.406152407
4       5        16777.577789481
4       3        53135.089988617
4       1        19343.755250369
6       5        8378.970491921
6       3        38056.782208702
6       1        52095.706509713
5       3        14857.910608473
5       1        25702.713214324
3       1        40153.580164158
```

Text file created by Conefor Inputs showing Euclidean distances between core areas.

3.3 Running the toolbox

Note: It's best not to display any outputs in ArcMap while running Linkage Mapper, because it may need to overwrite the outputs. If you get schema lock or permission errors, you may need to close ArcCatalog and ArcMap and start fresh without any output files displayed.

Click the *Linkage Mapper* toolbox, and click on *Build Network and Map Linkages*. The following dialog should appear. Lettered items are described below.



Linkage Mapper

Model Inputs

Project Directory

Core Area Feature Class

Core Area Field Name

Resistance Raster

Process Steps

☑ Step 1 - Identify Adjacent Core Areas

Adjacency Method: Cost-Weighted & Euclidean

☑ Step 2 - Construct a Network of Core Areas

Conefor Distances Text File

Network Adjacency Method: Cost-Weighted & Euclidean

☐ Step 3 - Calculate Cost-Weighted Distances and Least Cost Paths

☑ Drop Corridors that Intersect Core Areas

☐ Step 4 - Refine Network Using Options Below (optional)

Option A - Number of Connected Nearest Neighbors: 4

Option B - Nearest Neighbor Measurement Unit: Cost-Weighted Distance

☑ Option C - Connect Neighboring Constellations

☐ Step 5 - Calculate, Normalize and Mosaic Corridors

Additional Options

Bounding Circles Buffer Distance (optional)

Maximum Cost-Weighted Corridor Distance (optional)

Maximum Euclidean Corridor Distance (optional)

Linkage Mapper

Linkage Mapper is a GIS tool designed to support regional wildlife habitat connectivity analyses. It consists of several Python scripts, packaged as an ArcGIS toolbox, that automate mapping of wildlife habitat corridors. The scripts were developed for the Washington Wildlife Habitat Connectivity Working Group's 2010 statewide connectivity analysis.

A. **Input Data**
   a. **Project Directory:** Enter the project directory for your analyses (described in section 3.2 above). You should create a different project directory for each unique project/scenario you run, to avoid confusing data from different analyses.
   b. **Core Area Feature Class:** This is your map of core habitat area polygons (the polygons you will be connecting with linkages).

      Note: if you choose a feature class that is open in ArcMap and any of the features are selected, Linkage Mapper will only operate on those features.
   c. **Core Area Field Name:** Field must consist of positive integers < 9999 that identify unique core areas in your core area polygon file.
   d. **Resistance Raster:** Enter the resistance raster for your project.

B. **Check the steps you want to complete.** You can check all steps to run from start to finish in one shot. You can also re-start failed or killed runs at any step in the process by checking only those steps you want to re-do. **Note:** Step 4 is optional. Check step 4 and fill in step 4 options if you want to only connect each core to its 1-4 nearest neighbors and then connect clusters of cores together, rather than connecting all adjacent cores. Refer to Section 4 below for more details on what each step does.

C. **Step 1 - Identify Adjacent (neighboring) Core Areas**

   a. **Adjacency Method:** Specify whether to *calculate* adjacency data in Euclidean and/or cost-weighted distance space. Euclidean is fastest if you are doing quick draft products.

D. **Step 2 - Construct a Network of Core Areas**

   a. **Conefor Distances Text File:** Enter the distances text file generated by the Conefor Inputs tool.

   b. **Network Adjacency Method:** Specify whether to create links between core areas that are adjacent in Euclidean and/or cost-weighted distance space. You must have calculated adjacency data for your choice in step 1. If you choose both, corridors will be mapped between core areas that are adjacent in Euclidean OR cost-weighted distance space.

E. **Step 3 - Calculate Cost-Weighted Distances and Least-Cost Paths**

   a. **Drop Corridors that Intersect Core Areas:** Recommended. This will take effect in step 3. See detailed description in Section 4 below.

F. **Step 4 - Refine Network**

   You can limit the number of corridors mapped to those needed to ensure that each core area is connected to its 1-4 nearest neighbors (and then optionally connect any disjunct 'constellations' of core areas below).

   Number of Connected Nearest Neighbors: If step 4 is checked, this option lets you choose how many nearest neighbors to connect each core area to. Any links that are not needed to connect each core to its *N* nearest neighbors will be dropped.

   Nearest Neighbor Measurement Unit: Choose whether to measure 'nearest' above in Euclidean or cost-weighted distance.

   Connect Neighboring Constellations: After links that are not needed to connect each core to its *N* nearest neighbors are dropped, this option allows you to re-add links connecting nearest pairs of core area 'constellations,' i.e., discrete clusters of neighboring core areas, until all constellations are connected.

G. **Step 5: Calculate, Normalize, and Mosaic Corridors**

H. **Additional Options:**

   a. **Bounding Circles Buffer Distance (optional):** If a value is entered, this will create bounding circles around pairs of core areas (recommended for speed; see detailed description in Section 4 below). Buffer distances are entered in map units, e.g. a 10 km buffer would be entered as 10000 if map units were in meters. Leave blank if you don't want to use bounding circles.

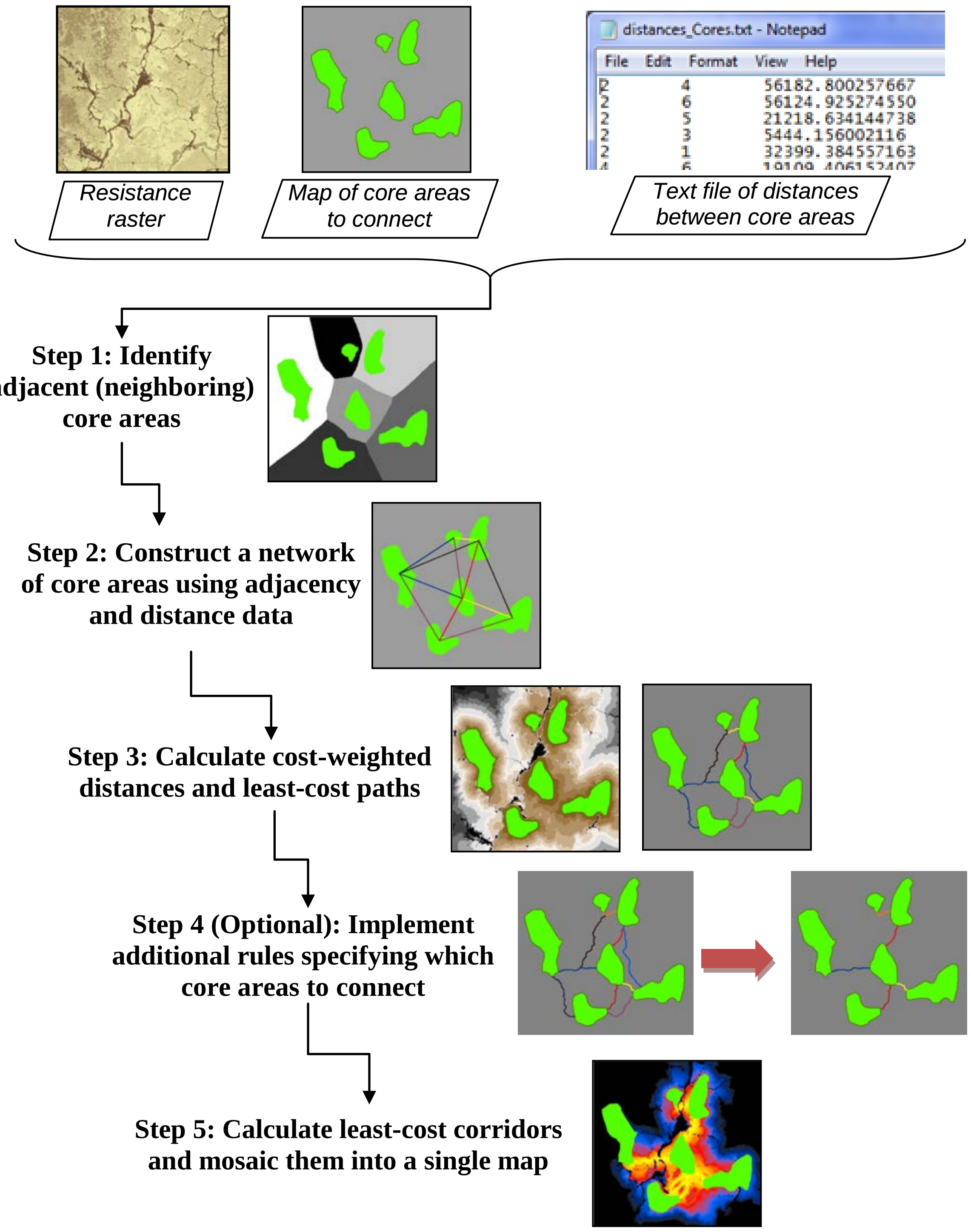
   b. **Maximum Corridor Distances:** Maximum distances can speed up calculations by automatically dropping excessively long corridors. If you are unsure, you should leave blank or overestimate. You can always make distances more stringent by re-running step 5. You can't make them less stringent without going back to step 2.

      i. **Maximum Cost-Weighted Corridor Distance:** If you only want to map corridors that are less than a maximum cost-weighted corridor length, enter it here (in map units). This will take effect in steps 3-5.

ii. **Maximum Euclidean Corridor Distance:** If you only want to map corridors between core areas that are closer than a certain distance apart, enter it here (in map units). This will take effect in steps 2-5.

4. What the Steps Do

As described above, the Build Network and Map Linkages tool takes 1) a core area polygon map, 2) a resistance raster, and 3) a text file specifying Euclidean distances between core area polygons. The tool then processes these data in 5 steps, illustrated and described below:

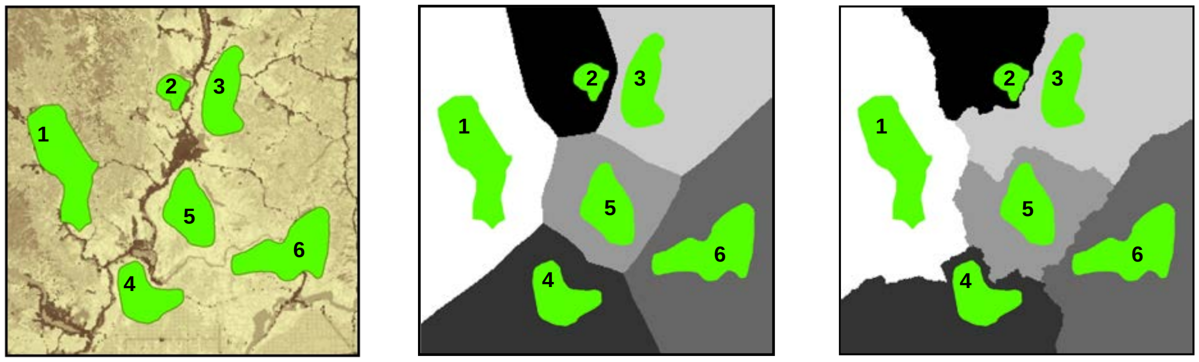

Resistance raster
Map of core areas to connect
distances_Cores.txt - Notepad
File Edit Format View Help
Text file of distances between core areas
Step 1: Identify adjacent (neighboring) core areas
Step 2: Construct a network of core areas using adjacency and distance data
Step 3: Calculate cost-weighted distances and least-cost paths
Step 4 (Optional): Implement additional rules specifying which core areas to connect
Step 5: Calculate least-cost corridors and mosaic them into a single map

```
2        4        56182.800257667
2        6        56124.925274550
2        5        21218.634144738
2        3        5444.156002116
2        1        32399.384557163
4        6        19109.406152407
```

Step 1: Identify adjacent (neighboring) core areas

Inputs: core area polygon file and resistance raster file.
Outputs: text files listing adjacent core areas in Euclidean (straight-line) and/or cost-weighted distance space (*euc_adj.csv* and/or *cwd_adj.csv*).

Step 1 identifies adjacent core areas using the ArcGIS ***Cost Allocation*** and ***Euclidean Allocation*** functions. Using your core area polygons and resistance raster, ArcGIS creates raster files that 'allocate' grid cells to the nearest core area in either Euclidean space or cost-weighted distance space. If a pathway from one core area to another must pass through the allocation zone of a third, then the two core areas are considered nonadjacent. For example, core areas *1* and *6* below are not adjacent because it is impossible to move from one to the other without at least passing through the allocation zone of core areas *2, 4*, or *5*. You can specify that core areas must be adjacent in Euclidean **or** cost-weighted distance space. This step creates the *'adj'* subdirectory within your project directory and stores rasters with data on adjacency and distances there. Adjacency lists are stored in text files *euc_adj.csv* and *cwd_adj.csv* within your *datapass* directory.

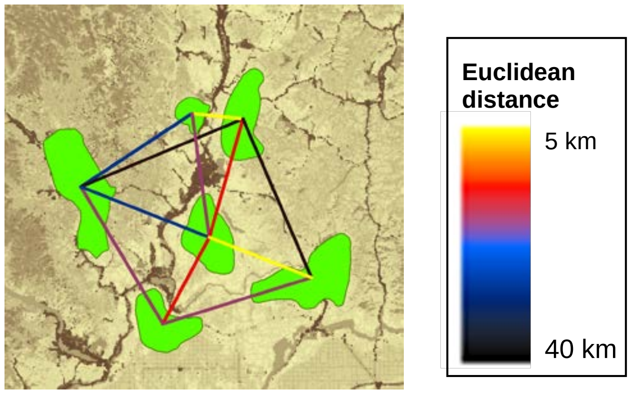


Core areas, resistance surface, and allocation grids for Linkage Mapper tutorial data. Left: core areas (green) and resistance surface (lower resistances shown in lighter shades). Center: Euclidean allocation zones, with different shades for each core area. Right: cost-weighted distance allocation zones. In this example, core areas *1* and *3* are adjacent in cost-weighted distance space, but not in Euclidean space.

Step 2: Construct a network of core areas using adjacency and distance data

Inputs: core area polygon file, text file of distances produced by Conefor Inputs, and core area adjacency files (*euc_adj.csv* and/or *cwd_adj.csv*).
Outputs: text file describing network (*linkTable_s2.csv*), and stick map showing links (*sticks_s2.shp*).

This step recognizes the text files specifying core area adjacency (*euc_adj.csv* and *cwd_adj.csv*) created in step 1, as well as the distances calculated using the Conefor Inputs tool. It uses these to create a "stick map" (*sticks_s2.shp*) connecting core area pairs that are candidates for corridor mapping. This step also generates *linkTable_s2.csv,* a table viewable in Microsoft Excel. This table describes the network and keeps track of which links are active, which are dropped, and statistics on each. These files are all stored in your *datapass* directory, and are used and updated with new versions (*linktable_s3.csv*, etc.) in subsequent steps. They are also saved in your *log* directory for reference. This step allows you to automatically discard links that are longer than a specified maximum Euclidean distance.



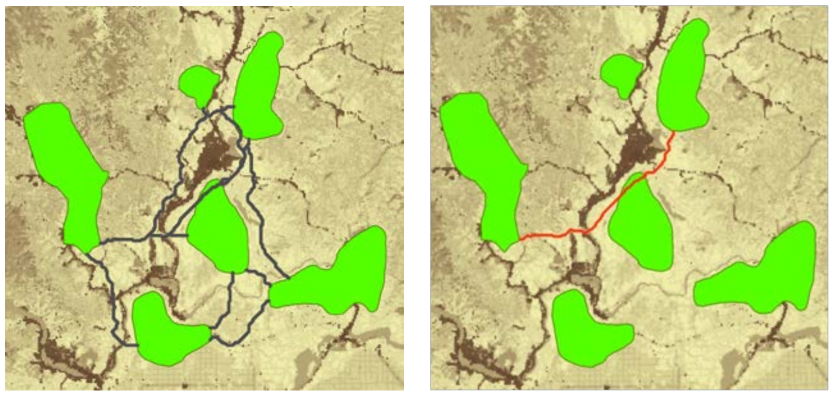
“Stick Map” created in step 2 (stored in *log* directory as *sticks_s2.shp*). Stick numbers show potential links between adjacent core areas, and include information on each. In this example, stick colors correspond to Euclidean (straight-line) distances between polygon edges.

Step 3: Calculate cost-weighted distances and least-cost paths

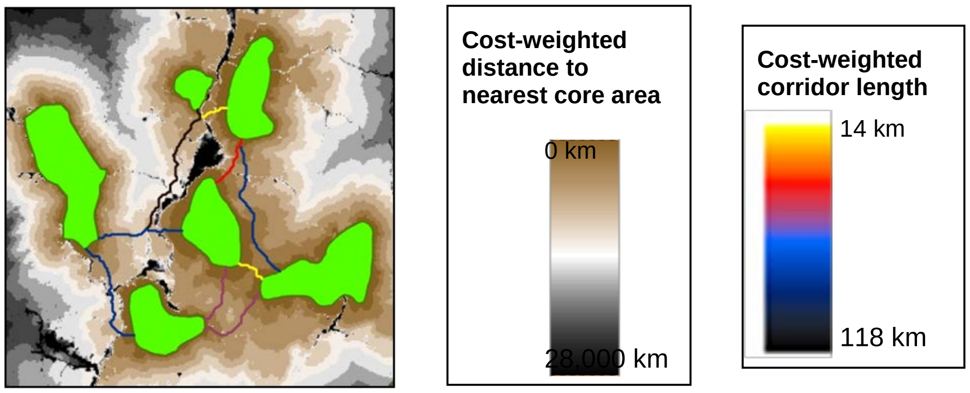
Inputs: core area polygon file, resistance raster file, *linkTable_s2.csv*.
Outputs: cost-weighted distance rasters for each core area (stored in *cwd* directory), a single cost-weighted distance raster for all core areas (*cwd*, stored in the *output* directory), *linkTable_s3.csv, sticks_s3.csv, lcplines_s3.csv*.

Step 3 uses the core area polygons, resistance raster, and link table from step 2 to perform cost-weighted distance calculations from each core area. As each cost-weighted distance surface is created, Linkage Mapper also extracts minimum cost-weighted distances between source and target core area pairs.

Step 3 also maps least-cost paths (the route along which the least resistance is accumulated) from core to core, using the cost-weighted distance raster and a direction raster for each core area. This step allows users to automatically discard linkages where the least-cost path passes through an intermediate core area (see graphic below) or linkages that are longer than a specified maximum cost-weighted or Euclidean distance.



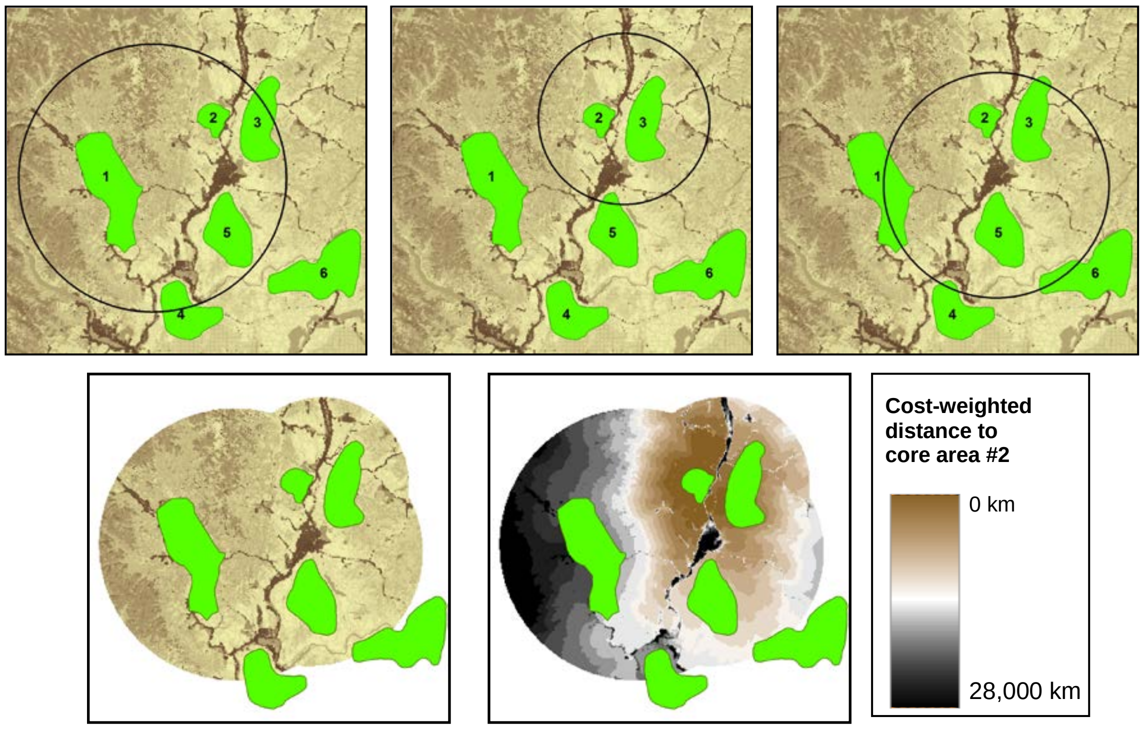
Least-cost paths between adjacent core areas (left). The red path (right) passes through an intermediate core area- that link would be dropped if the Drop Corridors that Intersect Core Areas box were checked. Links that are too long in Euclidean or cost-weighted distance can also be dropped automatically.


Cost-weighted distance to nearest core area
0 km
28,000 km
Cost-weighted corridor length
14 km
118 km

Step 3 places *cwd***,** a raster showing the cost-weighted distance to the nearest core area, in the *output* directory. Least-cost path lines are attributed with various linkage statistics (e.g. the total cost-weighted distance traversed by the linkage, shown above).

Minimizing processing time using the bounding circles option

Using bounding circles can considerably speed up calculations. Linkage Mapper first calculates bounding circles around "source" and "target" core area pairs. The resistance layer is then clipped by the union of all bounding circles for the source core area plus a buffer large enough to allow corridors sufficient room to "roam." For example, the graphic below shows buffer distances of 10 km (entered as 10000 map units under the Buffer Distance for Bounding Circles in the Additional Options section of the toolbox). This limits cost-weighted distance calculations from each source core to include only the portion of the landscape likely to be relevant to connectivity between the sources and target cores, reducing processing time.



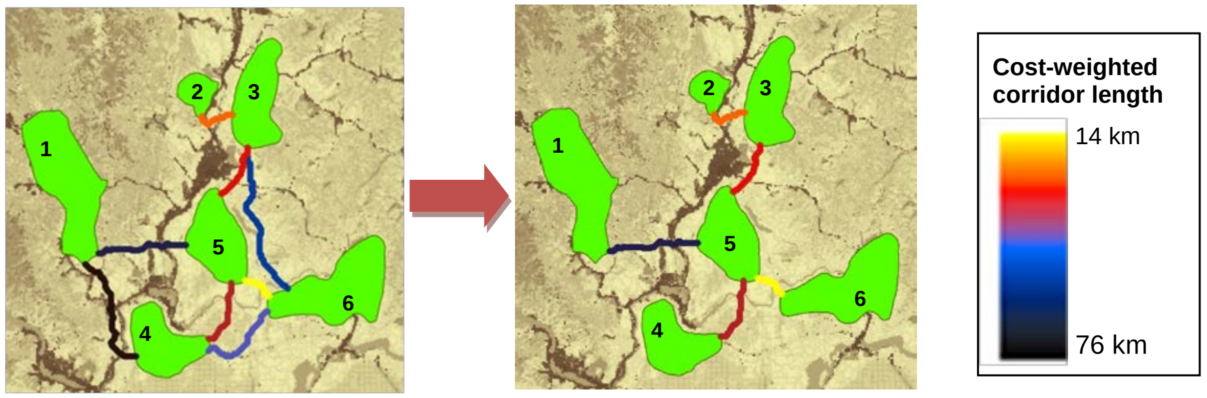
Bounding circles calculated for core area 2. Top: circles encompass core area *2* and each respective 'target' core area (*1*, *3*, and *5*) plus a buffer distance. Bottom left: for cost-weighted distance calculations from any core area, the algorithm extracts the portion of the resistance layer that falls within the union of all circles containing that core area. Bottom right: cost-weighted distance calculated from core area *2*.

Note: If you're unsure of an appropriate buffer distance, we recommend using a large value or skipping entirely. However, be aware that for large datasets with hundreds of core areas, the processing time can add up fast.

Step 4: Implement optional rules specifying which core area to connect

Inputs: *linkTable_s3.csv, lcplines_s3.shp.*
Outputs: *linkTable_s4.csv, sticks_s4.shp, lcplines_s4.shp.*

Step 4 optionally lets you connect each core area to just its 1-4 nearest neighbors, and then connect disjunct clusters (constellations) if desired. The latter will connect clusters of core areas together, starting with the closest, until all clusters are connected (as long as they are within maximum corridor lengths, if specified). You can measure 'nearest' in either Euclidean or cost-weighted distance.



Here's how step 4 alters the network when run with the option to connect each core area to its nearest neighbor (Step 4 option A = 1) in cost-weighted distance units and then connect disjunct constellations (Step 4 option C). In this case, two constellations were formed, with the two northernmost cores constituting one constellation and the remaining four cores constituting the other. The link between core areas *3* and *5* then connected the two constellations to form a single, connected network.

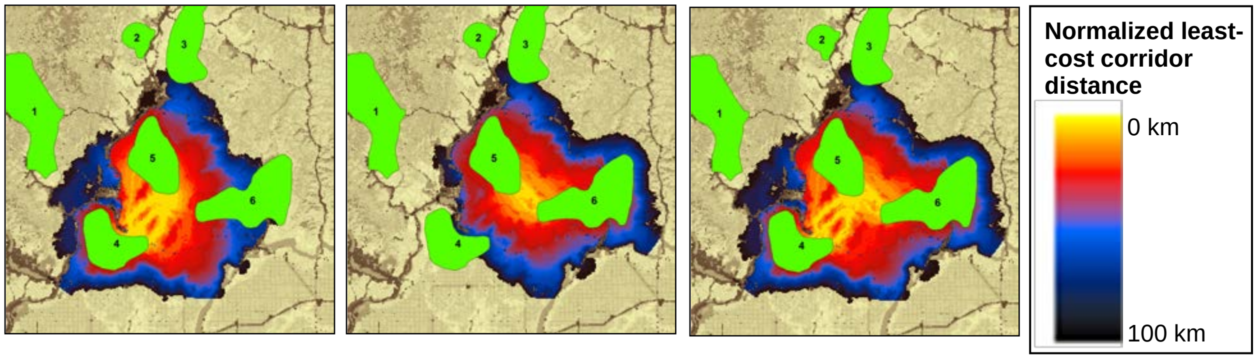
Step 5: Calculate least-cost corridors and mosaic them into a single map

Inputs: *linkTable_s4.csv or linkTable_s3.csv*; cost-weighted distance rasters; *lcplines_s4.shp* (if present in *datapass* directory) or lcplines_s3.shp.
Outputs: Final products including *linkTable_s5.csv*, (stored in the *output* directory), and final linkage maps and least-cost path feature classes (stored in *corridors.gdb* and *link_maps.gdb* in the *output* directory).

Step 5 calculates least-cost corridors (the sum of cost-weighted distance rasters calculated from each pair of core areas that are connected). It also normalizes least-cost-corridors by subtracting the least-cost path distance from the raw corridor:

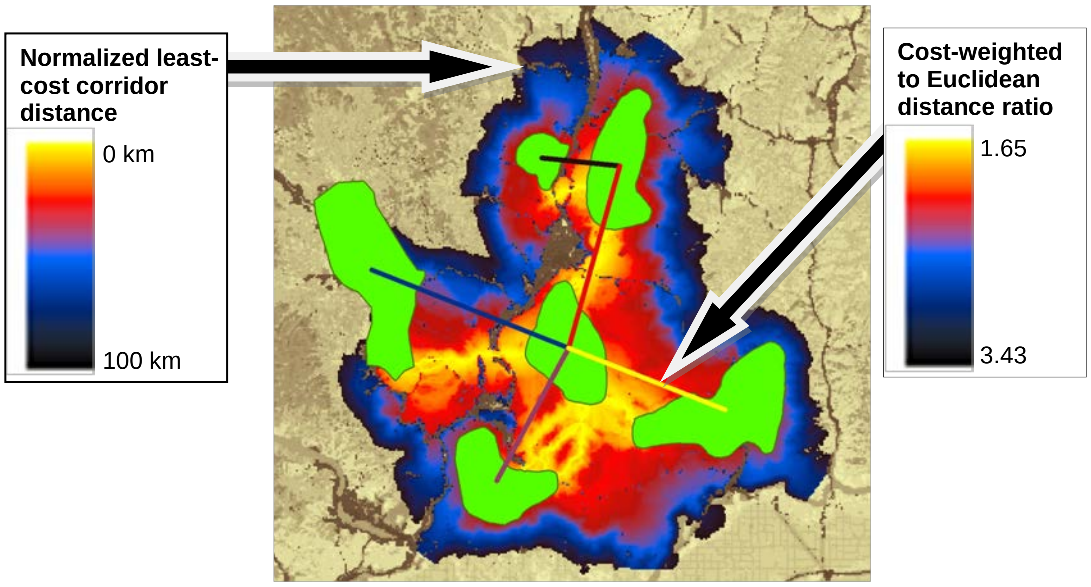
$$NLCC_{AB} = CWD_A + CWD_B - LCD_{AB}$$

Where $NLCC_{AB}$ is the normalized least cost corridor connecting core areas A and B, CWD_A is the cost-weighted distance from core area A, CWD_B is the cost-weighted distance from core area B, and LCD_{AB} is the cost-weighted distance accumulated moving along the ideal (least-cost) path connecting the core area pair. This step maps all corridors in the same 'currency;' grid cells in each normalized corridor raster range in value from 0 (the best or least-cost path) on up. Cell values are still in cost distance units, and reflect how much more costly the (locally optimal) path between the core areas passing through each cell is relative to the (globally optimal) least-cost path connecting the core area pair. As it calculates normalized corridors, step 5 combines them into a single map using the ArcGIS Mosaic function to create a composite linkage map in which each cell represents the minimum value of all individual normalized corridor layers.



Compositing process for normalized least-cost corridors. Panels show just two corridors as an example. Left and center panels show normalized least-cost corridors connecting core area *4* to core areas *5* and *6*, respectively. Colors indicate how much more costly the route between core area pairs passing through each cell is relative to the least-cost path connecting the core area pair. Linkage Mapper takes the minimum value of all normalized corridors to create a composite map (right panel).

This step also writes final stick and LCP maps, separated into active and inactive sticks and LCPs. Inactive links are those that have been dropped based on user criteria. It creates a final link table that only includes active links. Final link tables have extra columns with additional info, including *lcpLength* (the un-weighted length of the least-cost path), *cwdToEucRatio* (the ratio of cost-weighted distance to Euclidean distance between core areas), and *cwdToPathRatio* (the ratio of cost-weighted distance to the un-weighted length of the least-cost path, i.e. the distance traveled moving along the path). Finally, this step creates geodatabase (*link_maps.gdb and corridors.gdb*) in the *output* directory with final LCP and stick feature classes and mosaicked least-cost corridor maps.


Normalized least-cost corridor distance
0 km
100 km
Cost-weighted to Euclidean distance ratio
1.65
3.43

Normalized and mosaicked least cost corridors using the tutorial data (saved in linkages.gdb). Yellow grid cells are closer to corridor centers, with dark blue cells showing routes that accumulate up to 100km cost-weighted distance more than the optimal (least-cost) route. Stick colors show the ratio of cost-weighted distance to Euclidean distance for each link (cwd2Euc_r field in the feature classes), a metric of corridor quality. Yellow sticks show linkages that accumulate the least cost per unit Euclidean distance between core areas, while black sticks show linkages that accumulate the most.

5. Extra hints

5.1. 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 on how to do this in your version of ArcGIS.

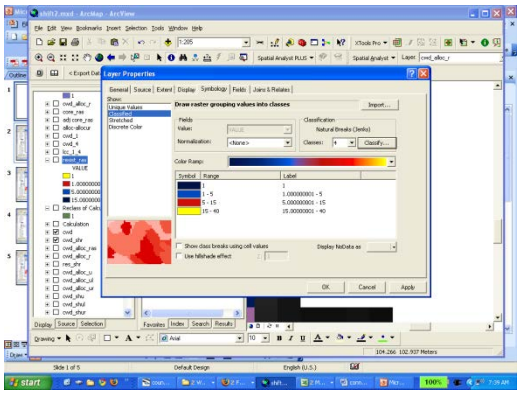
5.2. Manually adding and removing links

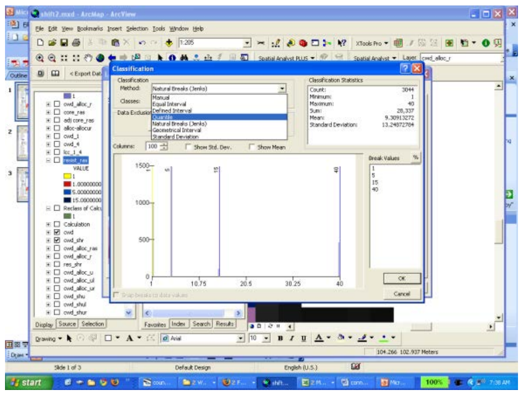
Links can be manually removed by editing one of the *linkTable* files in the *datapass* directory. For example, to remove a link before step 4 runs, edit *linkTable_s3.csv* and set the value for the link in the 'linkType' column to -100. Then run Linkage Mapper starting at step 4 by only checking Steps 4 and 5. Links with negative linkType values are automatically discarded, and -100 is reserved for user-removed links. We will soon add a feature to allow users to force retention of links.

5.3. Displaying Final Results

To color ramp your corridors:

In ArcGIS, double-click on the layer, and click the "Symbology" tab. You can used a "Stretched" renderer for quick display (often the Histogram Equalize stretch option works well). For more control of your color scheme, click "Classified" in the left-hand pane, and click "Classify" (see below left) above the color ramp.



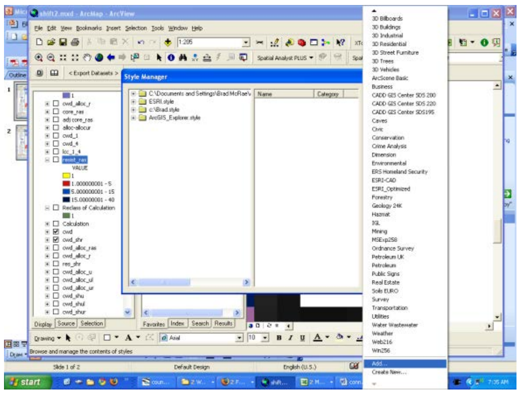


Choose 32 or more colors, and click "EQUAL INTERVAL" (see above right). You can also click "Exclusion" to focus your color ramp on relevant ranges of cost distance values (e.g., excluding areas much costlier than the optimal path). The Washington Connected Landscapes Project (WHCWG 2010) displayed only outputs up to a maximum linkage mapping cutoff distance from the ideal path.

Back in the main Symbology tab, choose a color scheme for your ramp by clicking on the horizontal color bar.

None of the basic ramps are great for coloring corridors, and you can actually create your own custom color scheme for ramps here. Or you can try the one we used for the Washington statewide analysis- it's included in the Linkage Mapper download as *Linkage Mapper.style* in the *toolbox/Styles* directory. To add it to your palette in ArcMap, click Tools >> Styles >> Style

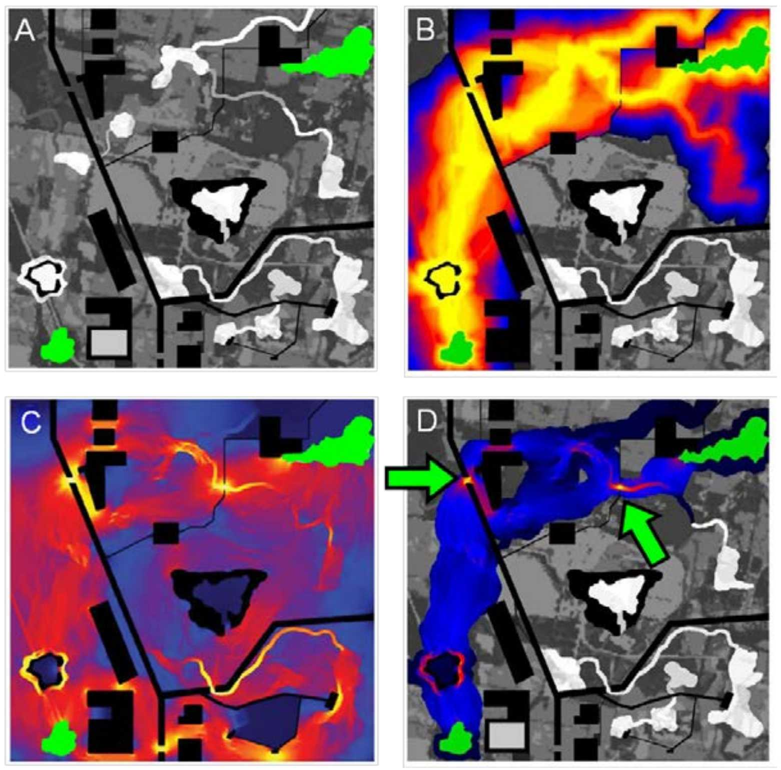
Manager. In Style Manager, go to the far right, click "styles" and scroll all the way to the bottom (see below). Click Add.



Now this color ramp should be in your palette.

5.4. Combining *Linkage Mapper* and *Circuitscape* to prioritize connectivity conservation

Circuit theory can complement least-cost analyses and help to prioritize important areas for connectivity conservation (McRae et al. 2008). Running Circuitscape (McRae and Shah 2009) within Linkage Mapper outputs is a way to get the best of both approaches, as illustrated below.



Example of how circuit theory can be used to identify and prioritize important areas for connectivity conservation. (A) Simple landscape, with two patches to be connected (green) separated by a matrix with varying resistance to dispersal (low resistance in white, higher resistance in darker shades, and complete barriers in black). (B) Least-cost corridor between the patches (lowest resistance routes in yellow, highest in blue). (C) Current flow between the same two patches derived using Circuitscape, with highest current densities shown in yellow (from McRae et al. 2008). Circuit analyses complement least-cost path results by identifying important alternative pathways and “pinch points,” where loss of a small area could disproportionately compromise connectivity. (D) A promising application is restricting circuit analyses to least-cost corridor slices to take advantage of the strengths of both approaches. This hybrid approach shows both the most efficient movement pathways and critical pinch points within them, which glow yellow. These could be prioritized over areas that contribute little to connectivity, such as the dark blue “corridor to nowhere” at the top right.

6. Support

Please join the Linkage Mapper User Group for updates. You may also use the issue tracker to report bugs and suggest enhancements. Both of these resources can be found on the Linkage Mapper website, where our code is hosted: http://code.google.com/p/linkage-mapper/

7. Literature Cited

Beier, P., W. Spencer, R. Baldwin, and B.H. McRae. 2011. Best science practices for developing regional connectivity maps. *Conservation Biology*.

McRae, B.H., B.G. Dickson, T.H. Keitt, and V.B. Shah. 2008. Using circuit theory to model connectivity in ecology, evolution, and conservation. *Ecology* 10: 2712-2724.

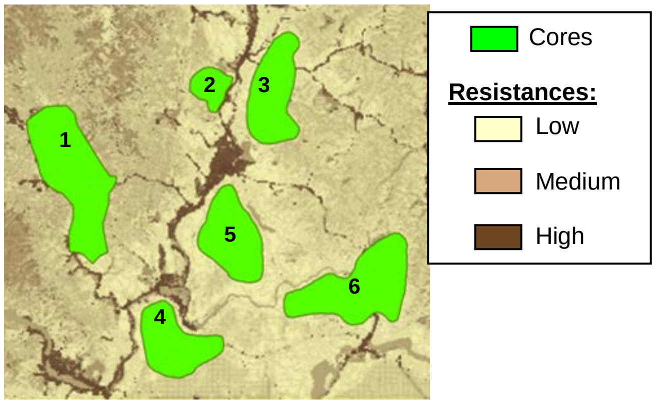
McRae, B.H., and Shah, V.B. 2009. Circuitscape User's Guide. ONLINE. The University of California, Santa Barbara. Available at: http://www.circuitscape.org.

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://www.waconnected.org.

8. Linkage Mapper Tutorial

We've provided a zipped directory that includes an ArcMap project **(*LM_Demo.mxd*)** along with two sample GIS layers:

- **A core areas shapefile (*cores.shp*):** Polygon shapefile defining habitat areas or natural landscape blocks to be connected. In this case, we're using data that are mostly fictionalized but loosely based on sharp-tailed grouse habitat it northeastern Washington.
- **A resistance raster (*resistances*):** A raster grid representing the difficulty or energetic cost of moving through each grid cell. Increasing values correspond to increasing movement difficulty.

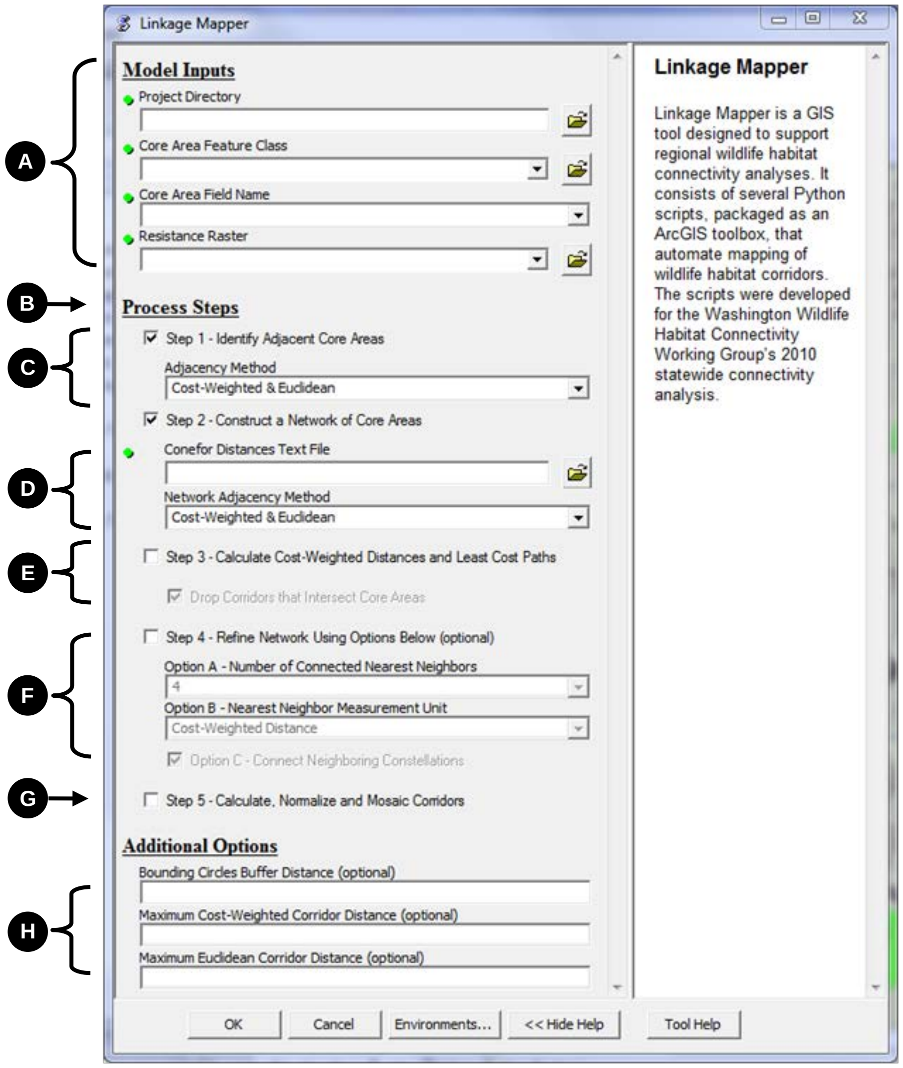


Demo data viewed in ***LM_Demo.mxd.***

1) Place the *LM_demo* directory somewhere easy to browse to *with no spaces or special characters in the directory name* (e.g., *C:\LM_demo*). ***LM_demo\Project*** will be your **project directory** for the tutorial.

2) Open LM_Demo.mxd in ArcMap.

3) Open the *Linkage Mapper* Toolbox, and click on *Build Network and Map Linkages*. The following dialog will appear:



Linkage Mapper

Model Inputs

Project Directory

Core Area Feature Class

Core Area Field Name

Resistance Raster

Process Steps

☑ Step 1 - Identify Adjacent Core Areas

Adjacency Method: Cost-Weighted & Euclidean

☑ Step 2 - Construct a Network of Core Areas

Conefor Distances Text File

Network Adjacency Method: Cost-Weighted & Euclidean

☐ Step 3 - Calculate Cost-Weighted Distances and Least Cost Paths

☑ Drop Corridors that Intersect Core Areas

☐ Step 4 - Refine Network Using Options Below (optional)

Option A - Number of Connected Nearest Neighbors: 4

Option B - Nearest Neighbor Measurement Unit: Cost-Weighted Distance

☑ Option C - Connect Neighboring Constellations

☐ Step 5 - Calculate, Normalize and Mosaic Corridors

Additional Options

Bounding Circles Buffer Distance (optional)

Maximum Cost-Weighted Corridor Distance (optional)

Maximum Euclidean Corridor Distance (optional)

Linkage Mapper

Linkage Mapper is a GIS tool designed to support regional wildlife habitat connectivity analyses. It consists of several Python scripts, packaged as an ArcGIS toolbox, that automate mapping of wildlife habitat corridors. The scripts were developed for the Washington Wildlife Habitat Connectivity Working Group's 2010 statewide connectivity analysis.

4) Enter the following (refer to letters in dialog image above):

A. *Input Data*

Project Directory: Browse to where you placed the **LM_demo** directory and select the **demoProject** directory within it.

Core Area Feature Class: choose **Cores**

Core Area Field Name: choose **core_ID**

Resistance Raster: choose **resistances**

B. **Check all the steps.**

C. *Step 1 - Identify Adjacent (neighboring) Core Areas*

Adjacency Method: Choose **Cost-weighted and Euclidean**.

D. *Step 2 - Construct a Network of Core Areas*

Conefor Distances Text File: Browse to where you placed the **LM_demo** directory and open the ***demoProject*** directory within it. Choose **distances_cores.txt.**

Network Adjacency Method: Choose **Cost-Weighted & Euclidean.**

E. *Step 3 - Calculate Cost-Weighted Distances and Least-Cost Paths*

Drop Corridors that Intersect Core Areas: **Check this box.**

F. *Step 4 - Refine Network*

Number of Connected Nearest Neighbors: Choose **1**.

Nearest Neighbor Measurement Unit: Choose **Euclidean distance.**

Connect Neighboring Constellations: **Check this box.**

G. *Step 5: Calculate, Normalize, and Mosaic Corridors*

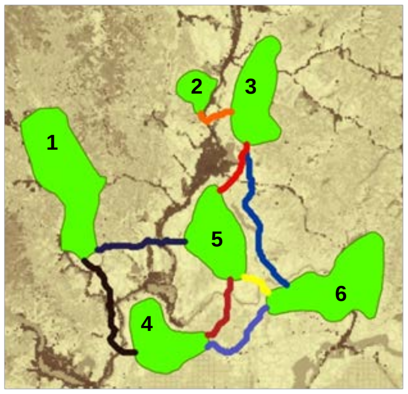
H. *Additional Options:*

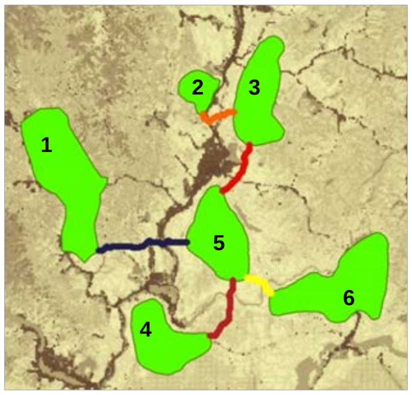
Bounding Circles buffer Distance (optional): Enter **10000** (equal to 10km).

Maximum Cost-Weighted Corridor Distance: Enter **100000** (equal to 100 km).

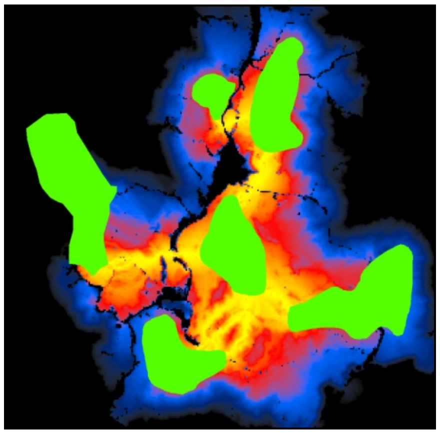
Maximum Euclidean Corridor Distance: Enter **40000** (equal to 40km).

Click OK. You can view outputs in ***LM_Demo_Results.mxd.***





Above left is the network showing active links after step 3 using the settings above (intermediate shapefile outputs are stored in the *log* directory). Yellow corridors have the lowest cost-weighted distances, and black corridors have the longest. Note that the corridor between cores *1* and *2* has been dropped because it is too long (118 km in cost-weighted distance units). The reason links have been dropped can be determined by querying the stick or lcp shapefiles. Above right is the network after running step 4. Each core area is connected to its nearest neighbor, and the link between cores *3* and *5* connects two otherwise disjunct constellations of core areas.



Above are normalized and mosaicked least-cost corridors (demoProject_lcc_mosaic_int in corridors.gdb) produced by the settings above.