**Linkage Priority Developer Documentation**

*Version 3.0—Updated October 2020*

John Gallo1, Randal Greene2and Darren Kavanagh3

1Conservation Biology Institute

2Feaver’s Lane

3Torilis, LLC

**Table of Contents**

[1 Introduction 2](#_Toc52964084)

[2 Coding Conventions 2](#_Toc52964085)

[3 LP Code Organization 2](#_Toc52964086)

[3.1 lp\_settings.py 2](#_Toc52964087)

[3.2 lp\_main.py 3](#_Toc52964088)

[3.3 Summary Diagram of all the modules 3](#_Toc52964089)

[3.4 Linkage Mapper.tbx 3](#_Toc52964090)

[3.5 LpDlgContent.xsl 3](#_Toc52964091)

[4 Geoprocessing Summary 5](#_Toc52964092)

[5 LP Functions 7](#_Toc52964093)

[5.1 lp\_main.py 7](#_Toc52964094)

[6 LM Architecture 10](#_Toc52964095)

[6.1 Configuration and Settings 10](#_Toc52964096)

[6.2 Logging 10](#_Toc52964097)

[6.3 Other Utilities 10](#_Toc52964098)

[7 Other Notes 11](#_Toc52964099)

[7.1 Source Control Using GitHub 11](#_Toc52964100)

[7.2 Debugging 11](#_Toc52964101)

[7.3 Becoming a Linkage Mapper Contributor 11](#_Toc52964102)

[7.4 Future Enhancements 11](#_Toc52964103)

# Introduction

Linkage Priority (LP) is an ArcGIS geoprocessing script tool, written in the Python programming language, that is included as part of the Linkage Mapper (LM) toolbox. Developers should first understand LM and LP functionality from the user’s perspective. Please review the following user-oriented introductory material:

* Linkage Mapper User Guide, including the LM tutorial.
* Linkage Priority User Guide, including the LP tutorial. Pay particular attention to the Geoprocessing Overview for high-level introduction to the processing logic that underlies LP.
* Linkage Mapper web site (<https://circuitscape.org/linkagemapper>) and related publications (<https://circuitscape.org/pubs>).

Developers should also be familiar with the ArcGIS geoprocessing framework on which LM and LP are built. For an introduction to this framework please refer to ESRI’s Geoprocessing Help. For ArcMap the URL is <https://desktop.arcgis.com/en/arcmap/latest/analyze/main/what-is-geoprocessing.htm> and for ArcGIS Pro the URL is <https://pro.arcgis.com/en/pro-app/help/analysis/geoprocessing/basics/what-is-geoprocessing-.htm>.

# Coding Conventions

As a Python project, developers are encouraged to follow the PEP8 style guide (<https://www.python.org/dev/peps/pep-0008/>). Please note the following conventions:

* Class, method, function and variable names – lowercase with underscore separators (e.g. function\_name()).
* Constant “variable” names – all uppercase (e.g. CONSTANT).
* Maximum line length – 79 characters.

# LP Code Organization

ArcGIS is flexible with regards to the installation location of custom toolboxes like Linkage Mapper. However, Linkage Mapper expects all Python files (files with .py extension) to be in a subdirectory called “scripts”.

Most of the LP functionality is defined in two Python modules – lp\_main.py and lp\_settings.py. The LP ArcGIS tool calls and passes the input parameters to lp\_main.py.

## lp\_settings.py

A module containing “constants” representing advanced LP settings/parameters that are not included in the tool dialog.

## lp\_main.py

The primary LP module, which includes:

* The execution starting point (if \_\_name\_\_ == "\_\_main\_\_ "), which calls the main() controlling function.
* The main() controlling function.
* Functions for the preparation steps.
* The run\_analysis() function and numerous called functions, which implement the primary analysis logic. The called functions are organized as per the Geoprocessing Summary below.
* Local helper functions.
* All functions are described in the LP Classes and Functions section below.

## Summary Diagram of all the modules

* On the following page there is a summary diagram of all the modules in the Linkage Mapper toolbox (including those unrelated to Linkage Priority).
* It gives a good overview of dependencies.
* It was created using the Pyreverse package.

## Linkage Mapper.tbx

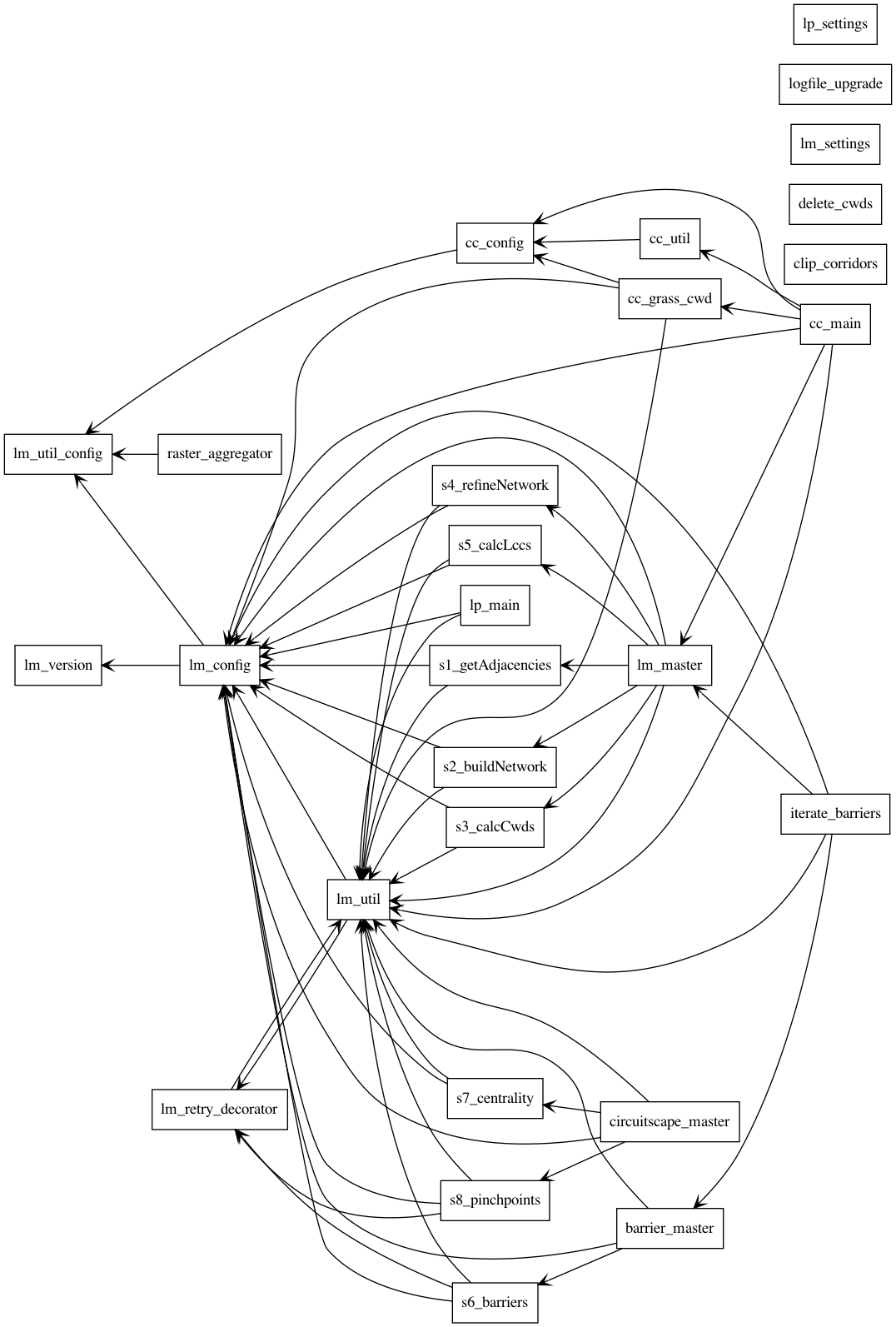
This file defines the Linkage Mapper ArcGIS toolbox. It is modified by right-clicking on it in ArcMap or ArcCatalog and selecting:

* Item Description, to view the tool metadata (including help text) and edit it.
* Properties, to define the tool’s:
  + Name
  + Label
  + Description
  + Optional underlying stylesheet
  + Underlying .py script file
  + Parameters (the type and order of these must match the order in which they are processed in the code)
  + Optional validation code to be run before the .py script is called
  + Optional compiled help file

In general, the toolbox should only be modified when there are new tools, new parameters for existing tools or updated help text.

## LpDlgContent.xsl

This file defines the stylesheet used by the Linkage Priority tool. It facilitates group labeling and nesting of parameters in the tool dialog. It only works for ArcGIS Desktop.



See note on previous page regarding this summary diagram.

# Geoprocessing Summary

The LP User Guide v2.0 included a geoprocessing overview. The geoprocessing was changed slightly between v2.0 and v3.0, and that section was not updated. That section is [here as a google doc](https://docs.google.com/document/d/1_NKZnXJ5ficCY60J4thxQtux27Z1yh3FTT2ZYS-TlQ0/edit#heading=h.2d8q8jupww1), and [here as a word document](https://docs.google.com/document/d/12SMXe8R1l-NtVoLlzCijd08TQIJ-BtgB/edit?usp=sharing&ouid=105337441051896869859&rtpof=true&sd=true). More up to date (we think) details are provided as follows:

* Check that LM in the same Project Directory successfully finished Steps 3 and 5, and terminate if issues are encountered
* Make preliminary calculations for each corridor
  + Calculate Permeability
    - In project\_LCPs line feature class, calculate the attribute Raw\_Perm as LCP\_Length / CW\_Dist
    - In project\_LCPs line feature class, calculate the attribute Rel\_Perm as a normalization of all Raw\_Perm values
  + Calculate Relative Closeness
    - In project\_LCPs line feature class, calculate the attribute Rel\_Close as a normalization of all LCP\_Length values
* Calculate Core Area Value (CAV) and its components for each core
  + Check weights and warn if issues
  + Add and calculate attributes in the input Core Area Feature Class
    - Mean resistance: mean\_res
    - Normalized resistance: norm\_res
    - Size: area
    - Normalized size: norm\_size
    - Perimeter: perimeter
    - Area/perimeter ratio: ap\_ratio
    - Normalized area/perimeter ratio: norm\_ratio
    - [Optional] Other core area value: ocav
    - [Optional] Normalized other core area value: nocav
    - [Optional] Expert core area value: ecav
    - [Optional] Normalized expert core area value: necav
    - [Optional] Current Flow Centrality value (from Centrality Mapper): CF\_Central
    - [Optional] Normalized CF\_Central value: ncfc
    - Core area value: cav
      * (norm\_res \* Resistance Weight) +

(norm\_size \* Size Weight) +\_

(norm\_ratio \* Area/Perimeter Weight) +

(necav \* Expert Core Area Value Weight) +

(ncfc \* Current Flow Centrality Weight) +

(nocav \* Other Core Area Value Weight)

* + - Normalized core area value: norm\_cav
* [Optional] Calculate Corridor Specific Value and Blended Priority raster
  + Calculate corridor specific priority (CSP) raster for each corridor
    - Check weights and warn if issues
    - Normalize Expert Corridor Importance Value: neciv
  + [Optional] Calculate Climate Linkage Priority Value for each core area
    - Current climate envelope: cclim\_env
      * [Optional] Calculate the mean values of a raster within each core area: fclim\_env
    - Calculate and save climate ratios to LCP layer and include start core and destination
    - Save Climate Analog Linkage Priority Value (A) and Climate Preference Linkage Priority Value (L) for each core pair to LCP Layer
    - Normalize climate priority values (A & L) for each core pair
    - Combine climate priority values A & L in a weighted sum to yield Core Areas Climate Linkage Priority Value (O): Clim\_Lnk\_Priority
  + Average the core area value of the two cores in the corridor: avg\_cav
  + Calculate CSP raster:
    - (Rel\_Close \* Closeness Weight) + (Rel\_Perm \* Permeability Weight) +

(avg\_cav \* Core Area Value Weight) + (neciv \* Expert Corridor Importance Value Weight) + (Clim\_Lnk\_Priority \* Climate Envelope Difference Weight)

* + Normalize CSP values
  + [Optional] Flag low quality corridors not to use
  + [Optional] Create blended priority raster
    - For each CSP raster
  + [Optional] Save a copy of Cores as the "Output for ModelBuilder Precondition"

# LP Functions

## lp\_main.py

**add\_output\_path**(in\_str)

Append LinkMap GDB path to inputted value.

**blended\_priority**(rast\_list, lcp\_ncsp)

Calculate overall Blended Priority.

**calc\_blended\_priority**(lcp\_lines)

Generate Blended Priority raster from NLCC rasters.

**calc\_cav**(core\_lyr)

Calculate Core Area Value (CAV) and its components for each core.

**calc\_closeness**(lcp\_lines)

Calculate relative closeness for each Least Cost Path.

**calc\_csp**(lcp\_lines, core\_lyr)

Calculate Corridor Specific Priority (CSP) for each linkage.

**calc\_permeability**(lcp\_lines)

Calculate raw and relative permeability for each Least Cost Path.

**check\_add\_field**(feature\_class, field\_name, data\_type)

Check if field exists, and if not then add.

**chk\_cav\_wts**()

Check weights used in CAV calculation.

**chk\_csp\_wts**()

Check weights used in CSP calculation.

**chk\_lnk\_tbls**()

Check that LM finished with steps 3 and 5.

**clim\_env\_read**(core\_lyr, core, field)

Read core climate envelope.

**clim\_envelope**(core\_lyr)

Determine Climate Envelope for each core.

**clim\_linkage\_priority**(lcp\_lines, core\_lyr)

Calculate Core Areas Climate Linkage Priority Value.

**clim\_lnk\_value**(xlnk, min\_pnt, target\_pnt, max\_pnt)

Calculate climate linkage value using equation of straight line.

**clim\_priority\_combine**(lcp\_lines)

Combine climate priority (A & L) values.  
   
Combine climate priority values A & L in a weighted sum to yield Core  
Areas Climate Linkage Priority Value (O).

**clim\_priority\_val\_normal**(lcp\_lines)

Normalize climate priority values (A & L) for each core pair.

**clim\_priority\_values**(lcp\_lines)

Save climate priority values for each core pair to LCP layer.  
   
Save Climate Analog Linkage Priority Value (A) and  
Climate Preference Linkage Priority Value (L) for each core pair  
to LCP Layer.

**clim\_ratios**(lcp\_lines, core\_lyr)

Calculate and save climate ratios to LCP layer.  
   
Calculate and save climate ratios to LCP layer and include start core

and destination.

**clip\_nlcc\_to\_threashold**(lcp\_list)

Clip NLCC\_A\_B rasters to CWD threshold.  
   
Clip the normalized least cost corridors using the specified CWD  
Threshold.

**core\_mean**(in\_rast, core\_lyr, in\_var)

Calculate the mean values of a raster within each core area.

**create\_run\_gdbs**()

Create scratch and if necessary intermediate GDB.

**eciv**()

Normalize Expert Corridor Importance Value (ECIV) for each corridor.

**get\_lcp\_fc**()

Get LCP feature class. Raise error if not found.

**get\_lm\_params**(argv)

Get settings from Linkage Pathways inputs.

**intercept**(point, slope\_val)

Find the intercept of a straight line.  
   
Where (x,y) is a point on the line and b is the slope of the line.

**inv\_norm**(rast\_list)

Invert and normalize each corridor.

**lcp\_csp\_for\_bp**(lcp\_lines)

Get list of LCPs and their normalized CSP values for BP raster

creation.  
   
Filter LCPs based on CPS cutoff if applicable. Return list with LCP  
filenames and dictionary with LCP GDB filename as key and normalized

CSP as value.

**log\_setup**()

Set up Linkage Mapper logging.

**main**(argv=None)

Run Linkage Priority tool.

**make\_core\_lyr**()

Create feature layer from cores feature class.  
   
Raise error if core feature class is not found.

**normalize\_field**(in\_table, in\_field, out\_field, normalization\_method='MAX\_VALUE', invert=False)

Normalize values in in\_field into out\_field.  
   
Normalize values in in\_field into out\_field using score range or max  
score method, with optional inversion.

**normalize\_raster**(in\_raster, normalization\_method='MAX\_VALUE', invert=False)

Normalize values in in\_raster.  
   
Normalize values in in\_raster using score range or max score method,  
with optional inversion.

**read\_lm\_params**(proj\_dir)

Read Linkage Pathways input parameters from log file.

**run\_analysis**()

Run main Linkage Priority analysis.

**save\_interm\_rast**(rast\_list, base\_name)

Save intermediate rasters if user chooses.

**sline\_y\_value**(x\_coord, slope\_val, intercept\_val)

For a x value on a straight line find its corresponding y value.  
   
The equation of a straight line is: y = mx + b  
where m is the slope of the line and b is the intercept.

**slope**(point1, point2)

Calculate slope of a straight line.  
   
Where (x1,y1) and (x2,y2) are points on the line.

**value\_range**(layer, field)

Get value range of field in layer.

# LM Architecture

As detailed above, LP modifies and builds on the standard LM outputs using the storage structure established by LM. As part of the LM family of tools, LP also takes advantage of some shared elements of the LM architecture.

## Configuration and Settings

For accessing the details of the successful LM run that LP builds on, LP instantiates and fills the lm\_config() class (defined in lm\_config.py) as lm\_env. This includes access to LM’s advanced settings defined in lm\_settings.py.

## Logging

LM provides logging to both the ArcGIS geoprocessing framework (for display and geoprocessing results history within ArcGIS) and to text files within the LM project structure (in the run\_history folder). LP uses the following functions logging functions from lm\_util.py:

**create\_log\_file**(messageDir, toolName, inParameters):

Creates and new text file for logging and remembers it for the current run.

**write\_log**(string):

Write the string to the current log file.

**close\_log**\_file():

Write the current time to the current log file and close the file.

**gprint**(string):

Write the string to the current log file and pass to the ArcGIS geoprocessing framework as a message or warning.

**raise\_error**(msg):

Write the message to the current log file, pass it to the ArcGIS geoprocessing framework as an error, and close the log file.

## Other Utilities

LP also uses the following functions from lm\_util.py:

**build\_stats**(raster):

Builds statistics and pyramids for output rasters.

**delete\_data**(dataset):

Delete the passed ArcGIS dataset.

# Other Notes

## Source Control Using GitHub

The LM family of tools are managed as an open source repository on GitHub. The repository URL is <https://github.com/linkagescape/linkage-mapper>.

## Debugging

For information on how to debug script tools in ArcGIS Desktop see ESRI’s ArcGIS *Debugging script tools* help page (<https://desktop.arcgis.com/en/arcmap/latest/analyze/creating-tools/debugging-script-tools.htm>) and for ArcGIS Pro see the *Debug Python code* help page (<https://pro.arcgis.com/en/pro-app/arcpy/get-started/debugging-python-code.htm>).

## Becoming a Linkage Mapper Contributor

We encourage contributions to the LM project by ArcGIS/Python developers. This could include enhancements and fixes to existing tools, and development of new tools for the LM toolbox. We encourage new tools to follow the protocols in Linkage Priority and Climate Linkage Mapper, which are currently the two newest tools in the LM toolbox.

## Future Enhancements

As with any active software project, there have been numerous suggestions for future enhancements. Please use the LM User Group (see the Support section of the LM User Guide) to register your suggestions.