

ArcGIS® 9

Geoprocessing Commands Quick Reference Guide



Copyright © 2004, 2005 ESRI
All rights reserved.
Printed in the United States of America.

The information contained in this document is the exclusive property of ESRI. This work is protected under United States copyright law and other international copyright treaties and conventions. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, or by any information storage or retrieval system, except as expressly permitted in writing by ESRI. All requests should be sent to Attention: Contracts and Legal Services Manager, ESRI, 380 New York Street, Redlands, CA 92373-8100, USA.

The information contained in this document is subject to change without notice.

Contributing Writers

Melanie Harlow, Ghislain Prince, Catherine Jones, Corey Tucker, Jeff Reinhart

U.S. Government Restricted/Limited Rights

Any software, documentation, and/or data delivered hereunder is subject to the terms of the License Agreement. In no event shall the U.S. Government acquire greater than RESTRICTED/LIMITED RIGHTS. At a minimum, use, duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in FAR §52.227-14 Alternates I, II, and III (JUN 1987); FAR §52.227-19 (JUN 1987) and/or FAR §12.211/12.212 (Commercial Technical Data/Computer Software); and DFARS §252.227-7015 (NOV 1995) (Technical Data) and/or DFARS §227.7202 (Computer Software), as applicable. Contractor/Manufacturer is ESRI, 380 New York Street, Redlands, CA 92373-8100, USA.

ESRI, ArcView, ArcGIS, ArcInfo, ArcCatalog, ArcToolbox, ArcSDE, ModelBuilder, ARC/INFO, ArcMap, 3D Analyst, ArcEditor, and www.esri.com are trademarks, registered trademarks, or service marks of ESRI in the United States, the European Community, or certain other jurisdictions.

Other companies and products mentioned herein are trademarks or registered trademarks of their respective trademark owners.






Table of Contents

Introduction	1
Analysis toolbox	5
Cartography toolbox	11
Conversion toolbox	13
Coverage toolbox	21
Data Management toolbox	39
Geocoding toolbox	67
Linear Referencing toolbox	69
Spatial Statistics toolbox	71
3D Analyst toolbox	79
Data Interoperability toolbox	93
Geostatistical Analyst toolbox	95
Network Analyst Toolbox	97
Spatial Analyst toolbox	101
Index	149
Appendix A: Toolset licensing	A-1

Introduction

This reference guide is designed to provide an easy and quick reference for those wishing to use the ESRI® command language at the ArcGIS® command line and for those writing scripts.

All commands are referred to as tools, scripts, or models and are maintained in toolsets within the ArcGIS toolboxes.

-  A toolbox can contain tools, toolsets, and scripts and is organized according to the collection of geoprocessing commands it contains.
-  A toolset can contain tools, toolsets, and scripts and is organized according to the geoprocessing commands it contains.
-  A tool is a single geoprocessing command.
-  A script is a set of instructions usually stored in a file and interpreted, or compiled, at run time.
-  A model consists of one process or, more commonly, multiple processes strung together.

This guide describes the following toolboxes:

Analysis toolbox

Cartography toolbox

Conversion toolbox

Coverage toolbox

Data Interoperability toolbox

Data Management toolbox

Geocoding toolbox

Geostatistical Analyst toolbox

Linear Referencing toolbox

Network Analyst toolbox

Spatial Analyst toolbox

Spatial Statistics toolbox

3D Analyst toolbox

Each toolbox contains a list of the toolsets and tools as they are organized within ArcToolbox™.

The Index section at the end of this guide contains an alphabetical list of each tool, script, toolset, and toolbox.

All tools are available with the ArcInfo™ license or the extension they are associated with. However, many are available for use with ArcView® or ArcEditor™ (sometimes with limited functionality). Those available with ArcView and ArcEditor are denoted with a ❖, and those available with ArcEditor are denoted with a ♦.

Some tools, such as Clip, exist in multiple toolboxes. Therefore, an alias can be added as a suffix to the tool name when more than one toolbox is available. An example of an alias usage is clip_arc, where clip is the tool and arc is the suffix representing the Coverage toolbox, or clip_analysis, where the suffix represents the Analysis toolbox.

The alias list:

Analysis toolbox	_analysis	Geostatistical Analyst toolbox	_ga
Cartography toolbox	_cartography	Linear Referencing toolbox	_lr
Conversion toolbox	_conversion	Network Analyst toolbox	_na
Coverage toolbox	_arc	Spatial Analyst toolbox	_sa
Data Interoperability toolbox	_di	Spatial Statistics toolbox	_stat
Data Management toolbox	_management	3D Analyst toolbox	_3d
Geocoding toolbox	_geocoding		

The syntax of an example tool:

```
Union_arc <in_cover> <union_cover> <out_cover> {fuzzy_tolerance} {JOIN | NO_JOIN}
```

Where:

Union_arc is the tool, and the components following it are the parameters

<> indicates required parameters.

{ } indicates optional parameters; these do not need to be included. One can be skipped using # if you need to apply only a portion of them.

The | indicates mutually exclusive arguments, and only one of the arguments in the list of options can be specified.

In some commands, there may be an ellipsis between two arguments, such as item1...item4. This indicates that you can give one or more (up to four in this example) names or values for that argument.

Example:

```
Union_arc Treepolycov Newtreecov Finaltreecov # JOIN
```

ArcGIS Desktop

Core geoprocessing tools



Analysis toolbox

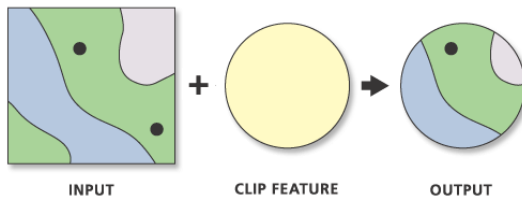
A suite of geoprocessing tools used to solve spatial or statistical problems.

Extract toolset

Contains tools used to manipulate data into manageable datasets containing only the desired features and attributes.

- ❖ **Clip:** extracts those features from an input feature class that overlap with features from a clip feature class.

`Clip <in_features> <clip_features> <out_feature_class> {cluster_tolerance}`



- The output feature class will have the attributes of the input features.
- The input features may be any geometry type, but clip features must have polygon geometry.

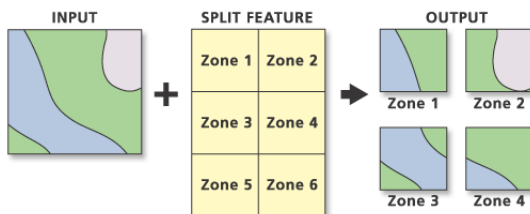
- ❖ **Select:** extracts selected features from an input feature class or layer and stores them in the output feature class.

`Select <in_features> <out_feature_class> {where_clause}`

- If no SQL expression is included, then all features will be included in the output feature class.
- If a SQL expression is used but returns nothing, the output feature class will be empty.

Split: clips the input features and stores them in multiple output datasets.

`Split <in_features> <split_features> <split_field> <out_workspace> {cluster_tolerance}`



- The split field data type must be character. The output feature classes will be named for split field values; therefore, they must start with a valid character.
- The number of output feature classes equals the total number of unique values in the split field.

- ❖ **Table Select:** extracts selected attributes from an input table or table view and stores them in an output table.

`TableSelect <in_table> <out_table> {where_clause}`

- The input can be an INFO® table, a dBASE® table, a geodatabase table, a VPF table, a feature class, or a table view.
- If a SQL expression is used but returns nothing, the output table will be empty.

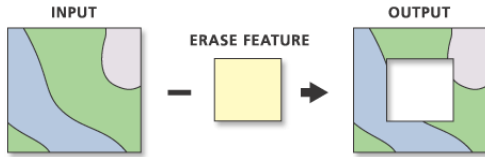


Overlay toolset

Contains tools for topological integration of features based on symmetry.

Erase: copies input features falling outside the erase polygon feature boundaries to the output.

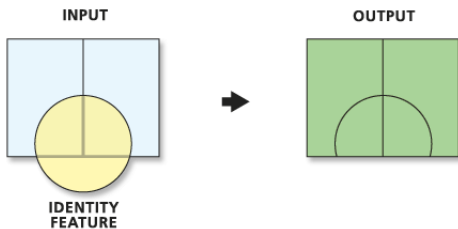
`Erase <in_features> <erase_features> <out_feature_class> {cluster_tolerance}`



- Input feature polygons that are coincident with erase feature polygons will be removed.
- The erase features must be polygons.

Identity: intersects two feature classes. The output contains the input features as well as those overlapping features of the identity feature class.

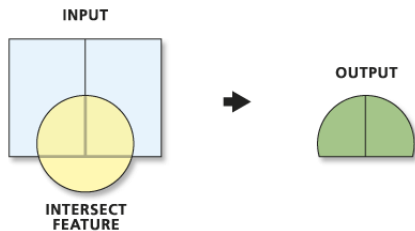
`Identity <in_features> <identity_features> <out_feature_class> {ALL | NO_FID | ONLY_FID} {cluster_tolerance} {NO_RELATIONSHIPS | KEEP_RELATIONSHIPS}`



- The input features must be point, multipoint, line, or polygon. The inputs cannot be annotation features, dimension features, or network features.
- The identity features must be polygons.

❖ **Intersect:** creates an output feature class containing features that fall within the area common to both input datasets.

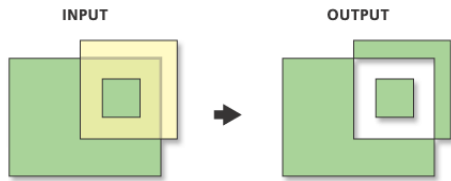
`Intersect <features{Ranks}; features{Ranks}...> <out_feature_class> {ALL | NO_FID | ONLY_FID} {cluster_tolerance} {INPUT | LINE | POINT}`



- The input features must be point, multipoint, line, or polygon. The inputs cannot be annotation features, dimension features, or network features.
- If the inputs have different geometry types (that is, line on poly, point on line, and so on), the output feature class geometry type will default to the same as the input features with the lowest dimension geometry.

Symmetrical Difference: creates an output feature class containing features or portions of features common to only one of the inputs.

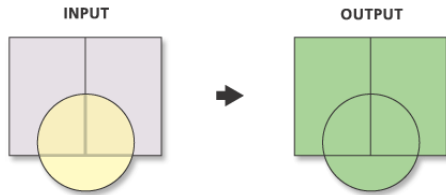
`SymDiff <in_features> <update_features> <out_feature_class> {ALL | NO_FID | ONLY_FID} {cluster_tolerance}`



- The input and difference feature class, as well as the output feature layer must have polygon geometry.

❖ **Union:** creates an output feature class containing all features from both inputs.

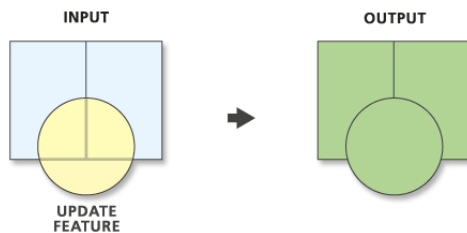
`Union <features {Ranks}; features {Ranks}...> <out_feature_class> {ALL | NO_FID | ONLY_FID}
{cluster_tolerance} {GAPS | NO_GAPS}`



- All input feature classes and feature layers must have polygon geometry.

Update: updates the attributes and geometry of the input using the update feature class or layer they overlap.

`Update <in_features> <update_features> <out_feature_class> {BORDERS | NO_BORDERS}
{cluster_tolerance}`



- The input features and update features must be of type polygon, and their names must match.

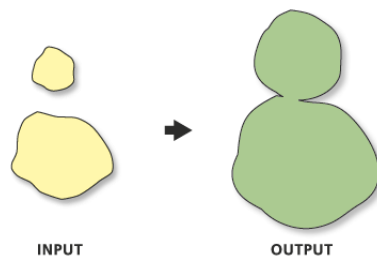


Proximity toolset


Contains tools to determine spatial relationships among features with respect to the distance relationships between features.

❖ **Buffer:** creates buffer polygons to a specified distance around the input features.

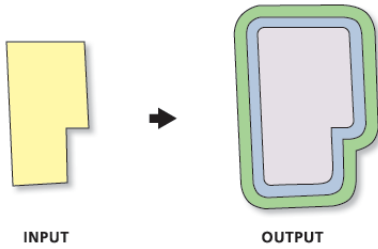
`Buffer <in_features> <out_feature_class> <buffer_distance_or_field> {FULL | LEFT | RIGHT}
{ROUND | FLAT} {NONE | ALL | LIST} {dissolve_field;dissolve_field...}`



- Features will not be buffered if their buffer distance is zero.
- Negative distances can be used when buffering polygon features to create buffers on the inside of the polygon features.

 **Multiple Ring Buffer:** creates a new feature class of buffer features using a set of buffer distances.

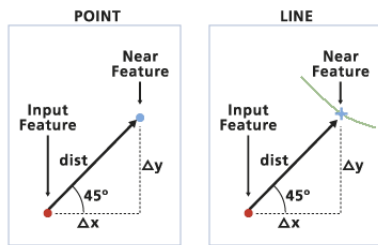
MultipleRingBuffer <input_features> <output_feature_class> <distances;distances...>
 {DEFAULT | CENTIMETERS | DECIMALDEGREES | FEET | INCHES | KILOMETERS | METERS | MILES |
 MILLIMETERS | NAUTICALMILES | POINTS | YARDS} {field_name} {ALL | NONE}



- If field name is not set, the default name for the field containing the distance value is distance. The field type is double.

Near: computes the distance from each point in the input to the nearest feature in the near feature class or layer.

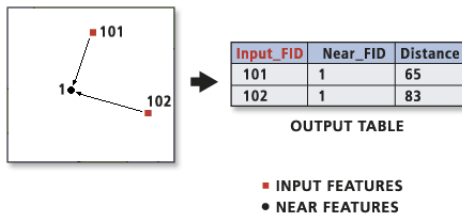
Near <in_features> <near_features> {search_radius} {NO_LOCATION | LOCATION}
 {NO_ANGLE | ANGLE}



- The results are recorded in the input features attribute table. Fields for distance and feature ID of the closest feature are added or updated. The field names are NEAR_DIST and NEAR_FID.

Point Distance: computes the distance between each point in a feature class or layer to all points in a different feature class or layer.

PointDistance <in_features> <near_features> <out_table> {search_radius}



- The results are recorded in an output table containing items for the feature's FID and DISTANCE. The field names are INPUT_FID, NEAR_FID, and DISTANCE.

Statistics toolset

Contains tools that perform standard statistical analysis on attribute data.

Frequency: calculates frequency statistics for one or more fields in a table.

Frequency <in_table> <out_table> <frequency_fields;frequency_fields...> {summary_fields;
 summary_fields...}

- The output table will contain the field frequency and the specified frequency fields and summary fields.

❖ **Summary Statistics:** calculates summary statistics for one or more fields in a table.

`Statistics <in_table> <out_table> <field{Statistic Type};field{Statistic Type}...>
{case_field}`

- The following statistical operations are available with this tool: sum, mean, maximum, minimum, range, standard deviation, first, and last. The median operation is not available.
- The Statistics Field(s) parameter Add Field button is used only in ModelBuilder™.

Cartography toolbox

Contains tools designed to produce data for maps to meet specific cartographic standards.

Masking toolset

Contains tools to construct masking polygons for use with variable depth masking.

Cul-de-Sac Masks: creates a feature class of polygon masks from a symbolized input line layer.

```
CuldeSacMasks <input_layer> <output_feature_class> <reference_scale> <spatial_reference>  
<margin> {ONLY_FID | NO_FID | ALL}
```

- This tool only accepts line layers as input.
- This tool only creates masks at the unconnected ends of lines in the input layer. These unconnected ends in the input layer are referred to as cul-de-sacs.
- A line end is considered connected if it shares its endpoint with the endpoint of another line.
- If the input line layer contains multipart line geometries, then cul-de-sac masks are created for all unconnected line ends, including the ends of parts within multipart lines.

Feature Outline Masks: creates mask polygons at a specified distance and shape around the symbolized features in the input layer.

```
FeatureOutlineMasks <input_layers> <output_feature_class> <reference_scale>  
<spatial_reference> <margin> <CONVEX_HULL | BOX | EXACT_SIMPLIFIED | EXACT>  
<ALL_FEATURES | ONLY_PLACED> {ONLY_FID | NO_FID | ALL}
```

- This tool accepts point, line, and polygon feature layers as well as geodatabase annotation layers as input.
- Margin values are specified in either page units or map units. Most of the time, you will want to specify your margin distance value in page units.
- If the input layer is an annotation layer, the reference scale will be automatically set to the reference scale of the layer's feature class to ensure accurate calculation of the mask.

Intersecting Layers Masks: creates masking polygons at a specified shape and size at the intersections of symbolized input layers.

```
IntersectingLayersMasks <masking_layer> <masked_layer> <output_feature_class>  
<reference_scale> <spatial_reference> <margin> <CONVEX_HULL | BOX | EXACT_SIMPLIFIED |  
EXACT> <ALL_FEATURES | ONLY_PLACED> {ONLY_FID | NO_FID | ALL}
```

- This tool accepts point, line, and polygon feature layers as well as geodatabase annotation layers as input.
- When creating masks, it is important to know that adding masks to maps adds complexity that will slow map drawing and affect map printing and exporting. Generally, there are three things to consider when creating masks for a map: (1) the number of masks, (2) the complexity of the masks, and (3) whether the masks will be used to mask polygon features filled with marker or line symbols. An increase in the number of masks, having more complex masks, and/or masking against marker or polygon fill symbols will result in slower drawing on your screen. In addition, printing and exporting performance can be poor and even fail to produce valid output, first, because of the large amount of processing required to print and export maps with masks, and second, because of known limitations in how graphic file formats store map export results that have many complicated masks.
- Masks will be created if the margin distance is 0 or negative. A margin size of 0 will create a polygon that represents the exact shape of the symbolized feature. A negative margin will result in a polygon smaller than the symbolized feature. Generally, a margin value larger than 0 will be specified to produce the desired masking effect.

Conversion toolbox

Contains tools that are used to convert data into various formats.

From Raster toolset

Contains tools to output raster datasets to other formats.

❖ **Raster To ASCII:** converts a raster dataset to an ASCII file representing raster data.

`RasterToASCII <in_raster> <out_ascii_file>`

- Both integer and floating-point rasters can be converted to an ASCII format.
- The NODATA_VALUE is the value in the ASCII file that will be assigned to the NODATA cells in the input raster. This value is normally reserved for those cells whose true value is unknown.
- The end of each row of data from the raster is terminated with a carriage return in the file.

❖ **Raster to Float:** converts a raster dataset to a file of binary floating-point values representing raster data.

`RasterToFloat <in_raster> <out_float_file>`

- The output will be a floating-point text file as an IEEE floating-point format, 32-bit signed binary file.
- Two outputs are created—an IEEE floating-point format, 32-bit signed binary file with a .flt extension and an ASCII header file named with a .hdr extension. Both will use the same Output floating-point raster file name.
- The NODATA_VALUE is the value in the output file assigned to those cells in the input raster that contain NoData. This value is normally reserved for those cells whose true value is unknown. By default, NoData values on the input raster will have a value of -9999 in the output float file.

❖ **Raster To Point:** converts a raster dataset to a point feature dataset.

`RasterToPoint <in_raster> <out_point_features> {raster_field}`

- For each cell of the input raster dataset, a point will be created in the output feature class. The points will be positioned at the centers of cells that they represent. The NoData cells will not be transformed into points.
- The input raster can have any cell size and may be any valid raster dataset.
- The feature output is assumed to be a shapefile.

❖ **Raster To Polygon:** converts a raster dataset to a polygon feature dataset.

`RasterToPolygon <in_raster> <out_polygon_features> {SIMPLIFY| NO_SIMPLIFY} {raster_field}`

- The input raster can have any cell size and may be any valid raster dataset.
- The Field parameter allows you to choose which column in the raster dataset will become an attribute in the output polygon file. The column containing the cell values (VALUE) will become a column with the heading Grid_code in the attribute table of the output feature class.

❖ **Raster To Polyline:** converts a raster dataset to a polyline feature dataset.

`RasterToPolyline <in_raster> <out_polyline_features> {ZERO | NODATA} {minimum_dangle_length} {SIMPLIFY| NO_SIMPLIFY} {raster_field}`

- The input raster can have any cell size and may be any valid raster dataset.
- The Field parameter allows you to choose which column in the raster dataset will become an attribute in the output polyline file. The column containing the cell values (VALUE) will become a column with the heading Grid_code in the attribute table of the output feature class.
- The feature output is assumed to be a shapefile.



To CAD Toolset

Contains tools to assist in the mining of CAD drawings for the purpose of populating new or existing geodatabase feature classes.

Add CAD Fields: adds fields to the input table by selecting from groups of CAD-specific fields, which have the appropriate name and type recognized by the Export To CAD tool.

```
AddCADFields <input_table> <ADD_ENTITY_PROPERTIES | NO_ENTITY_PROPERTIES>
{ADD_LAYER_PROPERTIES | NO_LAYER_PROPERTIES} {ADD_TEXT_PROPERTIES | NO_TEXT_PROPERTIES}
{ADD_DOCUMENT_PROPERTIES | NO_DOCUMENT_PROPERTIES} {ADD_XDATA_PROPERTIES |
NO_XDATA_PROPERTIES}
```

- If the input is a table view or a feature layer with a joined table, the fields are only added to the base table.
- Adding CAD fields to a feature class intended for export and calculating values into those fields is a quick way to specify the various CAD properties for export.
- It is useful to add Entity Property fields, Layer Property fields, Text Property fields, and CAD Document Property fields to separate tables to keep a normalized set of lookup tables that can be joined to express an organized CAD standard of how CAD files should be generated from feature class data.

Create CAD XData: creates a table formatted to be recognized by the Export To CAD tool as AutoCAD extended entity data.

```
CreateCADXData <in_table> <fields;fields...> <RegApp> <ADE | TRADITIONAL>
```

- The Export To CAD tool only recognizes extended entity data information linked to feature classes with a relationship class. To ensure that values in the table correspond directly with the feature attributes they were created from, it is suggested that the Add Relationship Class tool directly follow the Create CAD XData tool and have that followed with the Export To CAD tool.
- To attach extended entity data to exported features, an XData table-formatted file, similar to the one created with this tool, must be related to the output features, using a relationship class.
- It is important to remember that there must be a key field in the input and output tables so that the two tables can be related using the Create Relationship Class tool.

Export to CAD: exports features from a feature class to one or more CAD drawings.

```
ExportCAD <in_features;in_features...> <DWG-R2000 | DGN-V8 | DWG-R14 | DXF-R14 | DXF-R2000
| DWG-R2004 | DXF-R2004> <output_file> {USE_FILENAMES_IN_TABLES | IGNORE_FILENAMES_IN_
TABLES} {OVERWRITE_EXISTING_FILES | APPEND_TO_EXISTING_FILES} {Seed_File}
```

- This function is generally used as the final tool in the process of converting feature class data to new or existing CAD drawing files according to a predefined set of CAD drawing standards.
- If the CAD drawing files specified by the attributes of the exported features exist, CAD objects will be appended to those files. If the CAD files do not exist, the specified CAD files will be created with the standard defaults or using the optional seed drawing specified. If the CAD entity properties are not specified, then entities will be generated using the default entity properties of the drawing file or any existing layer symbology included in the optional seed drawing.
- To create cells or blocks with attributes, the definition for the cell or block must be defined in an existing target CAD drawing or in the seed drawing.
- This tool will not export coverage annotation to any CAD format.

Set CAD Alias: renames one or more existing field name aliases by matching columns from the input table with a list of CAD-specific fields of appropriate names, which are recognized by the Export To CAD tool.

```
SetCADAlias <input_table> <field_info>
```

- If a feature class intended for export already contains values useful for driving CAD properties, such as layer name, but the fields have different names, assigning a CAD field alias on that table using the Set CAD Alias tool is an efficient way to have the Export To CAD tool recognize those values as CAD properties.
- Shapefiles are not a valid input to this function, since they cannot maintain aliases for fields. If you need to use a shapefile as input, convert the shapefile to a layer file. Layer files or feature classes from a personal geodatabase or ArcSDE geodatabase are valid inputs to this tool.
- This tool overwrites the input, so be sure to make a backup of the original data.



To Coverage toolset

Contains tools to convert any supported feature class format to a coverage.

Feature Class To Coverage: creates a single coverage from one or more input feature classes or layers.

```
FeatureClassToCoverage <features{Type}; features{Type}...> <out_cover>
{cluster_tolerance} {DOUBLE | SINGLE}
```

- The cluster tolerance acts the same as the fuzzy tolerance in ArcInfo Workstation. The fuzzy tolerance of the output coverage will be the same as the cluster tolerance specified when executing this tool. If no cluster tolerance is specified, a default is calculated.
- It is suggested you run the Create Labels tool after successfully executing Feature Class To Coverage to ensure all polygon features have an accurate label.



To dBASE toolset

Contains tools to convert tables into a dBASE® format.

Table To dBASE: converts INFO™, OLE DB, or geodatabase tables to dBASE tables.

```
TableToDBASE <input_tables; input_tables...> <output_folder>
```

- The name of the output tables will be based on the name of the input table. To explicitly control the output name and for some additional conversion options, use the Table To Table tool.
- The Copy Rows and Table To Table tools can also be used to convert a table to a dBASE file.
- If the name of the output table already exists in the output folder, a number will be appended to the end to make it unique (for example, OutputTab_1.dbf).



To Geodatabase toolset

Contains tools to convert any supported vector or raster data type to a geodatabase.



Feature Class To Feature Class: copies a feature class to a geodatabase or to a shapefile.

```
FeatureClassToFeatureclass <input_features> <output_location> <output_feature_class_
name> {expression} {field_info} {SAME_AS_TEMPLATE | DISABLED | ENABLED} {SAME_AS_TEMPLATE
| DISABLED | ENABLED} {configuration_keyword} {first_spatial_grid}
```

- The Copy Features tool can also be used to convert a shapefile, coverage feature class, or geodatabase (personal or SDE) feature class to a shapefile or geodatabase (personal or SDE) feature class.
- Spatial indexes may be specified when creating an SDE or personal geodatabase feature class. The spatial index is used to quickly locate features that match the criteria of a spatial search. For most data, only a single spatial index is required.

- ❖ **Feature Class To Geodatabase (multiple):** copies one or more feature classes or layers to a geodatabase feature class.

`FeatureClassToGeodatabase <input_features;input_features...> <output_geodatabase>`

- The inputs can include shapefiles, coverage feature classes, VPF feature classes, or geodatabase feature classes. The inputs can also be feature layers.
- If the input is a layer with selected features, only those selected features will be written to the new output feature class.
- The name of the output feature classes will be based on the name of the input feature class name. For example, if the input is `c:\myworkspace\Gondor.shp`, the output feature class will be named `gondor`.
- If the name already exists in the output geodatabase, a number will be appended to the end to make it unique, for example, `_1`.

- ❖ **Import CAD Annotation:** converts a collection of CAD annotation features to a geodatabase annotation feature class.

`ImportCADAnnotations <input_features;input_features...> <output_feature_class>`
`<reference_scale> {CLASSES_FROM_LEVELS | ONE_CLASS_ONLY} {NO_MATCH | MATCH_FIRST_INPUT}`
`{NO_SYMBOL_REQUIRED | REQUIRE_SYMBOL} {STANDARD | FEATURE_LINKED} {linked_feature_`
`class} {AUTO_CREATE | NO_AUTO_CREATE} {AUTO_UPDATE | NO_AUTO_UPDATE}`

- Choose a reference scale that is roughly equal to the scale at which the annotation will normally be displayed. Based on this reference scale, symbols and text will appear larger as you zoom in on the annotation and smaller as you zoom out.
- The conversion requires an exclusive lock, so it may not be opened by another application. It also requires that if the annotation feature class is in an ArcSDE® geodatabase, it will not be registered as versioned.

- ❖ **Import Coverage Annotation:** imports coverage annotations into a geodatabase annotation feature class.


`ImportCoverageAnnotations <input_features;input_features...> <output_feature_class>`
`<reference_scale> {CLASSES_FROM_LEVELS | ONE_CLASS_ONLY} {NO_MATCH | MATCH_FIRST_INPUT}`
`{NO_SYMBOL_REQUIRED | REQUIRE_SYMBOL} {STANDARD | FEATURE_LINKED} {linked_feature_`
`class} {AUTO_CREATE | NO_AUTO_CREATE} {AUTO_UPDATE | NO_AUTO_UPDATE}`

- You can convert each coverage annotation level to individual annotation classes or merge them into a single class.
- The conversion requires an exclusive lock, so it may not be opened by another application. It also requires that if the annotation feature class is in an ArcSDE geodatabase, it will not be registered as versioned.
- If you select coverage annotation features and/or use a definition query, only those features that are selected and visible will be converted.
- You can create a selection set of coverage features and create a new layer from the selection. If you use that new layer as input to the conversion, only those features in the layer will be converted.

- ❖ **Import From CAD:** imports from one or more CAD files to a geodatabase.


`ImportCAD <input_files;input_files...> <out_personal_geodatabase> {spatial_reference}`
`{DO_NOT_EXPLODE_COMPLEX | EXPLODE_COMPLEX}`

- A fixed set of feature classes will be generated in the specified output feature dataset. These feature classes contain the geometry for the lines, areas, points, and document extent, and optionally, point feature classes can be generated for each unique block or cell name.
- This tool creates a new geodatabase and will not append to an existing one.
- CAD text and attribute entities are converted to point features.

 **Raster To Geodatabase (multiple):** loads multiple raster datasets into a geodatabase or raster catalog.


`RasterToGeodatabase <input_rasters;input_rasters...> <output_geodatabase> {configuration_keyword}`

- The output is the location of the geodatabase where you will store the raster.
- When converting the raster dataset to a personal geodatabase, the raster dataset is actually stored on the regular file system in a hidden folder.
- When converting the raster dataset to an ArcSDE geodatabase, the raster dataset is stored on the ArcSDE server as a raster ArcSDE format.

 **Table To Geodatabase (multiple):** converts dBASE, INFO, or OLE DB tables to geodatabase tables and copies tables from one geodatabase to another.

`TableToGeodatabase <input_table;input_table...> <output_geodatabase>`

- The inputs can include dBASE, INFO, VPF, OLE DB, or geodatabase tables. The inputs can also be table views.
- The name of the output table will be the same as the input.
- If a table's name already exists in the output geodatabase, a number will be appended to the end to make it unique (i.e. gondor_1).

 **Table To Table:** converts or copies dBASE, INFO, OLE DB, or geodatabase tables to a dBASE or geodatabase table.

`TableToTable <input_table> <output_location> <output_table_name> {expression} {field_info} {configuration_keyword}`

- The inputs can include dBASE, INFO, VPF, OLE DB, or geodatabase tables. The inputs can also be table views.
- To drop fields during the conversion, set their Field Info Visible property to FALSE. This will not affect the input table.
- If the input is a table view with a selection, only those rows that are selected will be transferred to the output.

To Raster toolset

Contains tools to convert any supported raster format to either a GRID, ERDAS IMAGE®, TIFF, or geodatabase format.

❖ASCII To Raster: converts an ASCII file representing raster data to a raster dataset.

`ASCIIToRaster <in_ascii_file> <out_raster> {INTEGER | FLOAT}`

- The ASCII file must consist of header information containing a set of keywords, followed by cell values in row-major order. The file format is:

```
<NCOLS xxx>
<NROWS xxx>
<XLLCENTER xxx | XLLCORNER xxx>
<YLLCENTER xxx | YLLCORNER xxx>
<CELLSIZE xxx>
{NODATA_VALUE xxx}
row 1
row 2
.
.
.
row n
```

where xxx is a number, and the keyword `nodata_value` is optional and defaults to -9999. Row 1 of the data is at the top of the grid, row 2 is just under row 1, and so on.

- The `nodata_value` is the value in the ASCII file to be assigned to those cells whose true value is unknown. In the raster, they will be assigned to NoData.
- Cell values should be delimited by spaces. No carriage returns are necessary at the end of each row in the grid. The number of columns in the header is used to determine when a new row begins.
- The number of cell values must be equal to the number of rows times the number of columns, or an error will be returned.

❖ **DEM To Raster:** converts a USGS DEM file to a raster dataset.

`DEMToRaster <in_dem_file> <out_raster> {FLOAT | INTEGER} {z_factor}`

- The resulting raster will have square cells. If the DEM has a different sample point spacing in the x and y directions, it is resampled during the conversion process. It is resampled using bilinear interpolation at a cell size equal to the smaller of the point spacings of the DEM in the x or y.
- For output to a grid raster, DEM To Raster transfers the projection and units information contained in the DEM header record to a map projection file in the output grid directory. If the output raster is not a grid, the projection information will be transferred to the `.aux` file.

❖ **Feature To Raster:** converts a feature dataset to a raster dataset.

`FeatureToRaster <in_features> <field> <out_raster> {cell_size}`

- Any shapefile, coverage, or geodatabase feature class containing point, line, or polygon features can be converted to a raster dataset.
- If the input field contains floating-point values, the output raster will be floating point; otherwise, it will be integer.
- The output cell size will determine the size of each pixel in the output raster dataset.

❖ **Float to Raster:** converts a file of binary floating-point values representing raster data to a raster dataset.

`FloatToRaster <in_float_file> <out_raster>`

- The input file is an IEEE floating-point format, 32-bit signed binary file.
- Two inputs are required: the binary floating-point file with a `.flt` extension (`<in_float_file>.flt`) and an ASCII header file with a `.hdr` extension (`<in_float_file>.hdr`). You only specify the `.flt` file; however, there must be an existing `.hdr` file in the same directory with the same file name.
- The ASCII file consists of header information containing a set of keywords. The file format is:

```
NCOLS xxx
NROWS xxx
XLLCENTER xxx | XLLCORNER xxx
YLLCENTER xxx | YLLCORNER xxx
CELLSIZE xxx
NODATA_VALUE xxx
BYTEORDER <MSBFIRST | LSBFIRST>
```

where xxx is a number, and the keyword `nodata_value` is optional.

❖ **Raster To Other Format (multiple):** converts one or more ArcGIS supported raster dataset formats to a GRID, IMAGINE, or TIFF format.

`RasterToOtherFormat <input_rasters;input_rasters...> <output_workspace>
{GRID | IMAGINE IMAGE | TIFF}`

- The input raster datasets can be any valid raster dataset that ArcGIS can recognize.

- This tool allows you to batch convert raster datasets to another raster dataset format. This is useful if you receive raster datasets in one format but you (or your client) prefer to use GRID, IMAGINE, or TIFF.



To Shapefile toolset

Contains tools to create shapefiles from feature classes.



❖ **Feature Class To Shapefile (multiple):** exports one or more feature classes to shapefiles in a designated folder.

`FeatureclassToShapefile <input_features;input_features...> <output_folder>`

- The name of the output shapefile will be the name of the input feature class. For example, if the input is c:\gdb.mdb\Gondor, the output shapefile will be named gondor.shp. To explicitly control the output shapefile name and for some additional conversion options, see the Feature Class To Feature Class tool.
- If the output shapefile already exists in the output folder, a number will be appended to the end to make it unique (for example, Gondor_1.shp).
- The coordinate system of each output shapefile will be the same as the input feature classes. If the Output Coordinate System Environment is set, the output will be projected to that coordinate system.

Coverage toolbox

Contains the original ArcInfo Workstation commands used to perform geoprocessing tasks with coverages.

Analysis toolset

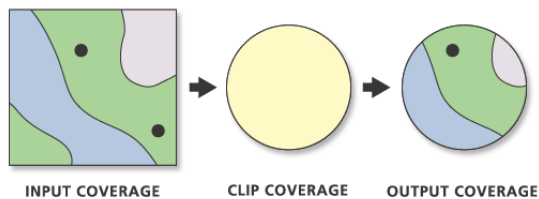
Contains tools and toolsets used for geospatial processing.

Extract toolset

Contains tools used to select features or parts of features to create a new coverage.

Clip: extracts, using a cookie cutter method, those features or portions of features from an input coverage that overlap with a clip coverage polygon.

`Clip <in_cover> <clip_cover> <out_cover> {POLY | LINE | POINT | NET | LINK | RAW}
{fuzzy_tolerance}`



- Clip maintains linear data belonging to different planar graphs in the same coverage. These may include arcs representing utility cables at different levels or a road passing over a stream. If there are arcs that appear to intersect but do not, nodes will not be inserted at the apparent intersection. Coincident and colinear line segments are preserved; additional vertices may be inserted. Two colinear arcs, one representing a road that follows the second—a stream—are maintained.
- The clip coverage must have polygon topology.
- Boundaries of interior polygons in the clip coverage are not used in Clip. Any clip coverage polygon whose internal number is greater than one is considered inside the clipping window.

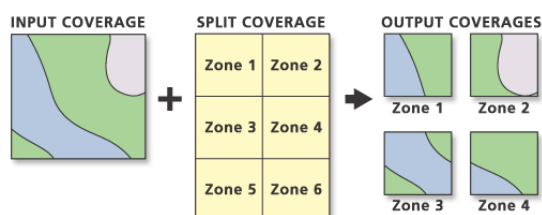
Select: extracts map features from the input coverage and stores them in an output coverage, based on logical expressions or by applying the criteria contained in a selection file.

`Reselect <in_cover> <out_cover> <info_express;info_express...> {POLY | LINE | POINT | ANNO.subclass | ROUTE.subclass | SECTION.subclass | REGION.subclass} {selection_file}
{out_feature_type}`

- When using the same input and output coverage for feature classes Anno, Section, Route, or Region, the output feature class subclass name must be different from the input feature class subclass name.
- Use of indexed items in the Query Builder can speed up the logical selection process. You can use the Index Item tool to create an attribute index.

Split: clips portions of the input coverage into multiple coverages.

`Split <in_cover> <split_cover> <split_item> <path> {POLY | LINE | POINT | NET | LINK | RAW}
{fuzzy_tolerance}`



- The output coverages will be named for split item values; therefore, they must start with a valid character.
- The split coverage must have polygon topology.
- The feature attribute table for each output coverage contains the same items as the input coverage feature attribute table.

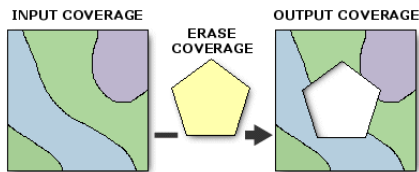


Overlay toolset

Contains tools used to calculate the various options when overlaying two coverages.

Erase: erases the input coverage features or portions of features that overlap with the erase coverage polygons.

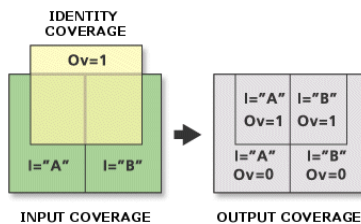
`Erase <in_cover> <erase_cover> <out_cover> {POLY | LINE | POINT | NET | LINK | RAW} {fuzzy_tolerance}`



- Input coverage polygons that are coincident with erase coverage polygons will be removed.
- User-IDs for all features will be the same in the output coverage as they are in the input coverage.
- Boundaries of interior polygons in the erase coverage are not used in ERASE. Any erase coverage polygon whose internal number is greater than one is considered inside the erasing window; an internal polygon number of one is considered outside. Only those input features (or portions of them) that are outside the erasing region are stored in the output coverage.

Identity: computes the geometric intersection of two coverages, where all features of the input coverage and only those overlapping from the identity coverage are preserved.

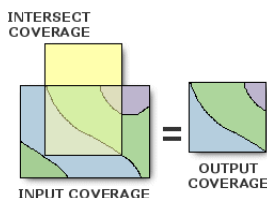
`Identity <in_cover> <identity_cover> <out_cover> {POLY | LINE | POINT} {fuzzy_tolerance} {JOIN | NO_JOIN}`



- The identity coverage must have polygon topology.
- Label points are generated in each output coverage polygon when the POLY option is used. The new polygon User-IDs are set equal to the polygon internal number minus one. When the LINE option is used, User-IDs of the input coverage are maintained.

Intersect: computes the geometric intersection of two coverages, where only those features in the area common to both coverages will be preserved.

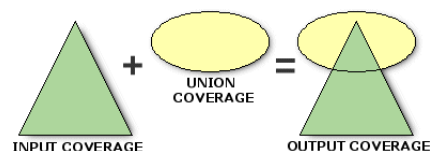
`Intersect <in_cover> <intersect_cover> <out_cover> {POLY | LINE | POINT} {fuzzy_tolerance} {JOIN | NO_JOIN}`



- The intersect coverage must have polygon topology.
- Label points are generated in each output coverage polygon when the POLY option is used. The new polygon User-IDs are set equal to the polygon internal number minus one.

Union: computes the geometric intersection of two polygon coverages. All polygons from both coverages will be split at their intersections and preserved in the output coverage.

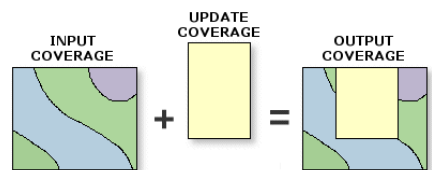
`Union <in_cover> <union_cover> <out_cover> {fuzzy_tolerance} {JOIN | NO_JOIN}`



- The input coverage and the union coverage must have polygon topology.
- Label points are generated in each output coverage polygon. The new polygon User-IDs are set equal to the polygon internal number minus one.
- Existing input coverage annotation is copied to the output coverage by Union.

Update: replaces the input coverage areas with the update coverage polygons using a cut-and-paste type of operation.

`Update <in_cover> <update_cover> <out_cover> {POLY | NET} {fuzzy_tolerance} {KEEP_BORDER | DROP_BORDER}`



- The input coverage and the update coverage must have polygon topology.
- New label point positions are generated for the output coverage polygons only when necessary. The User-ID for each polygon is equal to its old input coverage User-ID (the update coverage User-ID for updated polygons). Thus, you should attempt to make the User-ID values in the input coverage different from User-ID values in the update coverage to avoid having duplicate User-IDs in the output coverage.
- If DROP_BORDER is used, polygon boundaries along the outer edge of the update coverage are dropped. Even though the outer boundaries of some update polygons are dropped, the item values for the update polygons that overlap input coverage polygons will be assigned to the polygons in the output coverage. The DROP_BORDER option is not recommended for region coverages because of the possibility that some output regions may not be maintained.

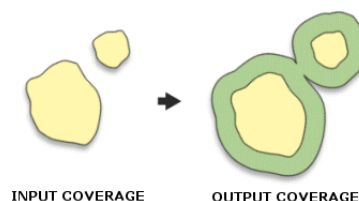


Proximity toolset

Contains tools used in geoprocessing analysis involving distance.

Buffer: creates buffer polygons around specified input coverage features.

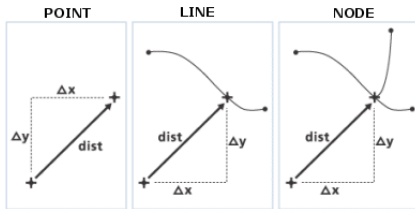
`Buffer <in_cover> <out_cover> {LINE | POLY | POINT | NODE} {buffer_item} {buffer_table} {buffer_distance} {fuzzy_tolerance} {ROUND | FLAT} {FULL | LEFT | RIGHT}`



- Negative and positive distances can be used for buffer distance with the POLY option. It is possible to shrink some polygons and grow others in the same coverage when the buffer item contains positive and negative numbers.
- The ROUND, FLAT, FULL, LEFT, and RIGHT options apply only to line data.
- The Buffer function works in Euclidean space and uses a two-dimensional algorithm. A buffer will be the same width no matter what the coordinate system is. It will not reflect the curvature or the shape of the earth. For the best results, generate the buffer in a map projection that minimizes distortion in the area of interest.

Near: computes the distance from each point in a coverage to the nearest arc, point, or node in another coverage.

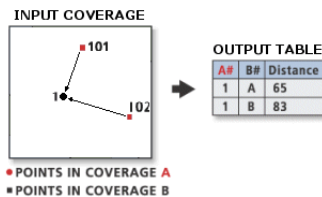
Near <in_cover> <near_cover> <out_cover> {LINE | POINT | NODE} {search_radius} {NO_LOCATION | LOCATION}



- Distance values are recalculated if this item already exists in the input coverage. If the Distance item is added, it will be in the same precision as the coverage.
- The calculated distance from point to arc will be from the point to the nearest location along the arc. The calculated distance from point to node will be between the nearest node locations on the arcs.

Point Distance: computes the distances between point features in one coverage to all points in a second coverage that are within the specified search radius.

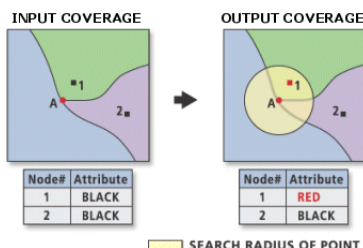
PointDistance <from_cover> <to_cover> <out_info_table> {search_radius}



- Distance is set to zero when no match is found within the search radius for a particular point. If no matching points are found, the tool gives a warning and no output INFO table is created.
- The output INFO table can become very large when both coverages contain many points. Use a smaller search radius to limit the number of combinations.
- The results are recorded in an output table containing items for the internal numbers and distance. The input with the highest precision for distance is the one used for the output INFO distance field.

Point Node: transfers attributes from a point feature class to a node feature class.

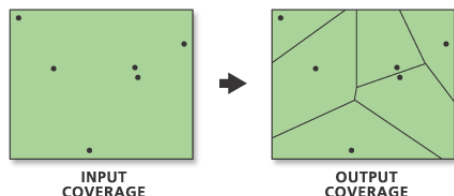
PointNode <point_cover> <node_cover> {search_radius}



- The coverage-ID number for each matching point is stored as the node-ID number in the NAT. If there are no matches to a node, then the node-ID is equal to the internal node number.
- The point cover must have a point attribute table for this command to work.
- The node cover can be the same as the point cover, in which case the attributes of the PAT are transferred to the NAT within the point coverage.

Thiessen: converts a point coverage to a coverage of Thiessen or proximal polygons.

Thiessen <in_cover> <out_cover> {proximal_tolerance}



- Thiessen polygons can be used to apportion a point coverage into regions known as Thiessen or Voronoi polygons. Each region contains only one Input Coverage point. Each region has the unique property that any location within a region is closer to the region's point than to the point of any other region.
- All items in the input coverage point attribute table (PAT) are copied to their associated polygons in the output coverage PAT.



Conversion toolset

Contains tools and toolsets to convert a coverage to or from another file format.



From Coverage toolset

Contains tools used to convert a coverage into various file formats.

Export To DLG: converts a coverage to an optional digital line graph (DLG-3) file format.

ArcDLG <in_cover> <out_dlg_file> {in_point_cover} {in_projection_file} {x_shift} {y_shift} {in_header_file} {TRANSFORM | NO_TRANSFORM}

- Before creating a digital line graph (DLG) file using Export To DLG, each node should be sequentially numbered using the Renumber Nodes tool. This will ensure that all arc, node, and polygon feature internal numbers are sequential.
- There are two distribution formats for a DLG file, standard and optional. This tool writes a DLG in the Optional format only.
- Coverage topology is saved in the DLG file using conventions that are similar to the way topology is stored in a coverage (for example, polygons are defined in clockwise loops, islands as counterclockwise loops, each feature has a unique identification number, negative numbers for lines indicate reverse directions, and so on).

Export To Interchange File: converts a coverage to an Interchange file.

Export <COVER | FONT | GRID | INFO | LINESET | MAP | MARKERSET | PLOT | SHADESET | STACK | STACKALL | TEXT | TEXTSET | TIN> <in_dataset> <interchange_file> {NONE | PARTIAL | FULL} {max_lines}

- When exporting a coverage, all associated INFO tables are written to the interchange file. For example, if the coverage name specified for input data is Forest, an INFO table named Forest.LABEL would be saved in the interchange file. A table named Forest1.LABEL, however, would not be saved in the interchange file.
- Export files created with the Compression option set to Full can be significantly smaller than export files created with the Partial or None options.

Export To S57: converts a coverage to an S-57 object format.

```
ArcS57 <in_workspace> <log_file> {out_workspace}
```

- S-57 is a data standard developed by the International Hydrographic Organization (IHO) to be used for the exchange of digital hydrographic data.
- The output log file will be created in the process, then holds the report of the export.

Export To SDTS: converts a coverage or grid to an SDTS topological vector profile (TVP) or point profile transfer file.

```
SDTSExport <TVP | POINT | RASTER> <in_dataset> <out_transfer_prefix> {in_point_cover}  
{out_DD_transfer} {conversion_control_file}
```

- SDTS is a large standard composed of smaller, more limited subsets that are federally approved as part of the SDTS FIPS 173 standard. These subsets are called profiles. The TVP (designed specifically for planar vector data with topology), raster, and point profiles are the only profiles supported by SDTSEXPORT.
- The following conditions must be met when creating a TVP transfer:
 - The coverage must have polygon topology.
 - The coverage cannot have a mask file; only clean coverages will export.
 - The coverage must have a projection defined.

Export To VPF: converts a coverage to either a VPF coverage or VPF tile.

```
VPFExport <in_cover> <out_file> {tile_name} {control_file} {EXTRA | NO_EXTRA} {NO_FIT | FIT}
```

- The coverage must not have a mask file. Use the Clean tool to remove mask files.
- A full VPF path name must be specified with output VPF coverage or table.
- The VPF standard specifies only coverages in geographic coordinates. Using units of decimal degrees, on the WGS 1984 datum, you cannot clean a coverage that has units in decimal degrees. You should build the coverage in this case or understand how cleaning will affect your coverage.

Ungenerate: Creates a text file of x,y coordinates from a coverage.

```
Ungenerate <in_cover> <out_generate_file> <LINE | POINT | POLY | TIC | LINK | REGION.subclass  
| ANNO.subclass> {NODES | NO_NODES} {EXPONENTIAL | FIXED}
```

- Ungenerate provides a useful mechanism to create simple coordinate files from coverages. This allows you to easily transfer coverages to other mapping systems or view and update individual coordinates using your computer's text editor.
- The coordinates created by Ungenerate are in the same coordinate precision as the input coverage. Single-precision coordinates are generated for single-precision coverages, and double-precision coordinates for double-precision coverages.



To Coverage toolset

Contains tools used to convert various file formats into coverages.

Advanced Tiger Conversion: performs the Basic Tiger Conversion, followed by advanced operations including joining, defining a projection, and building topology.

```
TigerTool <in_tiger_file_prefix> <out_cover_prefix> {NO_JOIN | JOIN} {UTM | STATE}  
{zone_number} {1995 | 1997 | 1998 | 1999 | 2000 | 2002} {NO_RESTART | RESTART}
```

- Advanced Tiger Conversion converts all versions released after April 1989.
- The Advanced Tiger Conversion tool does not support record types F and G released with the 1992 School District version. These are temporary record types not found in earlier or subsequent versions.

- The output coverages created in the TIGER file conversion will always be in double precision. TIGER/Line® files often contain tiny line segments that would be lost if converted to single precision.

Basic Tiger Conversion: converts U.S. Bureau of Census TIGER/Line® files to one or more coverages.

TigerArc <in_tiger_file_prefix> <out_cover> {out_point_cover} {out_landmark_cover}
{1995 | 1997 | 1998 | 1999 | 2000 | 2002}

- Basic Tiger Conversion converts all versions released after April 1989. The minimum input required by Basic Tiger Conversion is record types 1 and 2.
- The Basic Tiger Conversion tool does not support record types F and G released with the 1992 School District version. These are temporary record types, not found in earlier or subsequent versions.
- The output coverages for the Basic Tiger Conversion tool will always be in double precision. TIGER/Line files often contain tiny line segments that would be lost if converted to single precision.

Generate: creates a coverage from raw coordinates stored in a text file.

Generate <in_file> <out_cover> <LINES | ANNOTATIONS | CIRCLES | CURVES | FISHNET | LINKS | POINTS | POLYGONS | TICS>

- Generate creates new coordinate features but does not create topology or attributes for these features. Other tools, such as Build or Clean, can be used to create feature topology.
- The coordinate precision of the output coverage is determined by the precision setting. To convert a double-precision file to a double-precision coverage, the precision must be set to double.

Import From DLG: converts as standard or optional formatted digital line graph (DLG) file into a coverage.

DLGArc <in_dlg_file> <out_cover> {out_point_cover} {NOFIRST | ALL | ATTRIBUTED} {x_shift} {y_shift} {category}

- Topology data contained in the DLG file is ignored. You can use the Build tool after running Import From DLG, creating topology on the newly created coverage. Sometimes the coverage will have arc intersections and will need to be cleaned using the Clean tool.
- The output coverage may require editing before polygons or lines can be built and feature attribute tables created. For example, checks should be made on the output coverage to ensure that label points occur within their polygons, arcs match at nodes, polygons close, arcs do not cross, and so on.

Import From Interchange File: converts an interchange file to a coverage.

Import <AUTO | COVER | FONT | GRID | INFO | LINESET | PLOT | MAP | MARKERSET | SHADESET | STACK | TEXT | TEXTSET | TIN> <interchange_file> <out_dataset>

- IMPORT reads any export file that has been fully or partially compressed as well as decompressed. IMPORT automatically recognizes whether the export file is compressed.
- For the COVER option, all INFO data files saved in the interchange file whose names contain the coverage name prior to the last period in the INFO data file name are written to the workspace INFO database for the output coverage.

Import From S57: converts data from an S-57 file format to a coverage.

S57Arc <in_s57_file> <out_workspace> {CLEAN | NO_CLEAN}

- S-57 is a data standard developed by the International Hydrographic Organization (IHO) to be used for the exchange of digital hydrographic data.
- Each S-57 exchange dataset contains one catalog file and one or more base cells. Import From S57 reads the catalog file, converts it to an INFO file, then converts each base cell file to one or two ARC/INFO® coverages. One of these coverages will contain all the isolated nodes (for instance, spatial point objects); the other coverage will contain all the spatial and feature objects plus the data descriptive information.

- The Import From S57 importer creates either one or two ARC/INFO® coverages per base cell file, depending on the types of objects contained within the file.

Import From SDTS: creates coverages or grids from an SDTS topological vector profile (TVP) or point profile transfer file.

```
SDTSImport <in_transfer_prefix> <output> {out_point_cover} {layer_name} {DD | DROP_DD}
{PRESERVE | CONVERT}
```

- SDTS is a large standard composed of smaller, more limited subsets that are federally approved as part of the SDTS FIPS 173 standard. These subsets are called profiles. The TVP (designed specifically for planar vector data with topology), raster, and point profiles are the only profiles supported by Import From SDTS. The type of profile being converted is automatically determined by the command.
- Import From SDTS can read the U.S. Census Bureau TIGER, USGS DLG-3 vector data, National Geodetic Survey geodetic control point data, and USGS DEM raster data in SDTS format.
- Polygon and line topology is generated for TVP data. Point topology is generated for point profile data.

Import From VPF: converts a VPF table to an INFO table or converts a VPF coverage or tile to a coverage.

```
VPFImport <input_vpf> <output> {tile_name} {control_file} {NO_EXTRA | EXTRA}
```

- Full VPF path names must be specified with the input VPF coverage or table.
- If the VPF coverage was created using the Export To VPF tool with the option to convert all tables selected, then the output coverage will be identical to the input VPF coverage.



Data Management toolset

Contains tools and toolsets to manage, manipulate, and maintain coverages and their attribute tables.

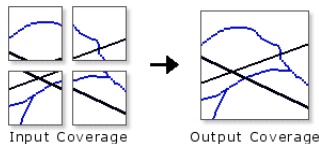


Aggregate toolset

Contains tools used to combine coverages.

Append: combines an unlimited number of coverages into one coverage.

```
Append <in_covers; in_covers...> <out_cover> {FEATURES_ONLY | FEATURES_ATTRIBUTES}
{POLY | LINE | POINT | NODE | NET | LINK | ANNO.subclass | SECTION.subclass | ROUTE.subclass
| REGION.subclass} {NO | TICS_ONLY | FEATURES_ONLY | FEATURES_TICS}
```



- All input coverages to be appended must contain the feature class or set of feature classes and feature attribute tables to be appended. For example, if the NET feature type option is used, all coverages should have line and polygon features and corresponding AATs and PATs.
- The item definitions of the feature attribute tables must be the same and in the same order for all appended coverages (unless the FEATURES_ONLY option is used).

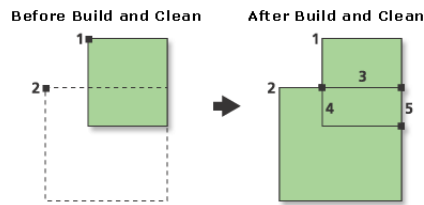


Composite Features toolset

Contains tools to create or convert regions within a coverage and to convert line features to routes.

Line Coverage To Region: converts arcs to preliminary regions in a new or existing coverage or appends preliminary regions to an existing region subclass.

```
RegionClass <in_cover> {out_cover} <out_subclass> {in_region_item} {out_region_item}
{selection_file} {MULTIRING | SINGLERING}
```



- The input coverage must have an AAT to specify the input region item.
- The arcs in each group, which are determined by the unique value of the input region item, must form closed loops. When the input region item is not specified, each arc in the input coverage becomes a preliminary region and should form a closed loop.
- If a selection file is not specified, all arcs are selected and available for grouping into regions. However, arcs in the input coverage that are already part of one or more fully structured regions are not available for appending to the subclass, since they may not form closed rings when grouped.

Line Coverage To Route: creates a route system by creating whole arc sections for each arc in the input coverage. It can also be used to append arcs to an existing route system.

`ArcRoute <in_cover> <out_route_system> {in_route_item} {out_route_item} {measure_item} {UL | UR | LL | LR} {BLANK | NO_BLANK}`

- Creates a route system from lines or appends lines to a route system. It groups lines that are topologically connected and have unique values for the input item to create the route system. The unique values of the Input item are always written to the output item in the route attribute table (RAT); these values help identify routes once they have been created.
- When appending routes to an existing route system, output route item must be the name of an existing item on the route attribute table of the route system. The tool will append a section to an existing route for every input arc having an input item equal to an output route item in the route attribute table, provided the input arcs are topologically connected to the route being appended. The measure item on the original part of the route being appended are updated based on the measures assigned to the new sections and the specified starting node. For those groups of arcs having values for the input route item not found in the output route item, a new route is created.

Polygon Coverage To Region: converts a polygon coverage to a region subclass. Each polygon in the in_cover becomes a region of the output subclass.

`PolyRegion <in_cover> <out_cover> <out_subclass>`

- Polygon Coverage To Region can be used on an input coverage that does not have arc topology; however, the input coverage must have polygon topology.
- The tool builds region topology for the output subclass. Topology in the input coverage is maintained in the output coverage.
- When the output coverage is the same as the input coverage, the subclass is created in that coverage.

Region To Polygon Coverage: converts a region subclass into a polygon coverage and creates an INFO table containing overlapping region information.

`RegionPoly <in_cover> <out_cover> <in_subclass> {out_table}`

- All items in the region subclass PAT are maintained in the output coverage PAT.
- The output coverage PAT contains only the attributes of the first region associated with each polygon. Values of zero indicate void areas in which the subclass does not exist.
- The attributes of the second to the nth regions associated with each polygon are stored in the output table.

- If only one region is associated with each polygon (a planar region subclass), then the output table does not need to be specified. However, an output table must be specified when using nonplanar region subclasses.



Generalization toolset

Contains tools to derive data with less detail and complexity from coverage features.

Aggregate Polygons: combines disjointed and adjacent polygons into new area features based on a distance.

`AggregatePolygons <in_cover> <out_cover> <cell_size> <distance> {NON_ORTHOGONAL | ORTHOGONAL}`



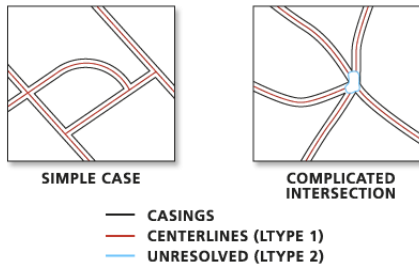
A) Nonorthogonal features

B) Orthogonal features

- This tool involves GRID functions and requires the ArcGIS Spatial Analyst extension software license.
- The input coverage must have a polygon topology.
- Because of the possibility of creating overlapping boundaries, preliminary regions are used as the resulting features. To create fully built regions from the preliminary regions, use the Clean tool with the POLY option on the output coverage.

Collapse Dual Lines To Centerline: derives centerlines (single lines) from dual-line features, such as road casings, based on specified width tolerances.

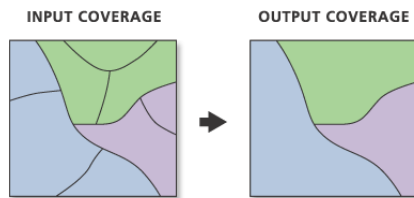
`collapseDualLineToCenterline <in_cover> <out_cover> <maximum_width> {minimum_width}`



- In addition to the standard items, the Output Coverage.AAT will contain the following five new items:
 - LTYPE—Contains a line type value of:
 - 1 centerlines
 - 2 unused lines and outlines of complicated intersections
 - 3 partition lines
 - LL#—Carries the left source arc record number.
 - RL#—Carries the right source arc record number.
 - L-ID—Carries the left source arc User-ID.
 - R-ID—Carries the right source arc User-ID.
- The values for item_width, output_width, and item_type in the item definition for all these items are 4, 5, and B.

Dissolve: merges adjacent polygons, lines, or regions that have the same value for a specified item.

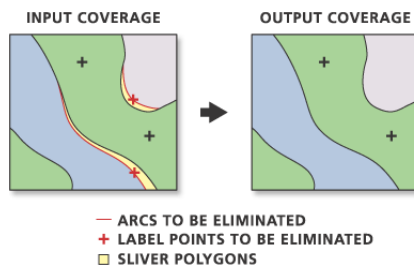
`Dissolve <in_cover> <out_cover> <dissolve_item> {POLY | LINE | NET | REGION.subclass}`



- Dissolve is used to create a simplified coverage from one that is more complex. Although the input coverage may contain information concerning many feature attributes, the output coverage contains information only about the dissolve item.
- The merging of polygons with Dissolve is the counterpart of intersecting polygons in overlays. Dissolve will remove the boundaries.
- With the POLY option, Dissolve will remove dangling arcs and pseudo nodes. The output coverage PAT with the POLY option or the output coverage AAT with the LINE option will only contain the dissolve item but no additional attributes. If #ALL is used as the dissolve item, then input coverage item definitions and data are preserved in the output coverage but User-IDs will be altered.

Eliminate: merges selected polygons with neighboring polygons that have the largest shared border between them or the largest area.

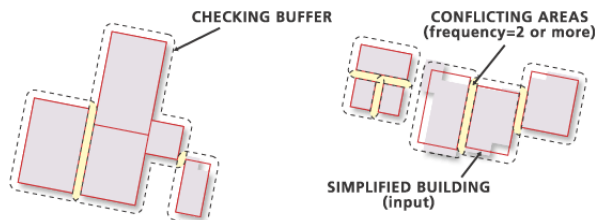
`Eliminate <in_cover> <out_cover> <info_express;info_express...> {NO_KEEP_EDGE | KEEP_EDGE} {POLY | LINE} {selection_file} {BORDER | AREA}`



- Only the selected set of polygons or lines will be eliminated. Polygons that border the background polygon will not be eliminated when KEEP_EDGE is specified.
- For the POLY option, an arc with a negative User-ID will never be eliminated, even if it's the longest arc in a selected polygon. When this happens, the next longest arc is eliminated unless it's along the coverage boundary when the Keep polygon boundary option is selected (KEEP_EDGE).
- Use of indexed items can speed up logical feature selection in Eliminate. See Index Item for details.

Find Conflicts: searches a region coverage for overlapping and closely spaced buildings, based on a specified distance, and records the occurrences.

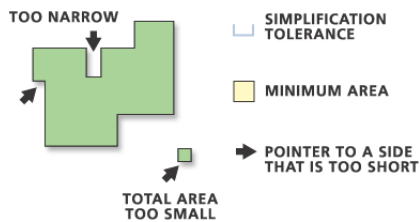
`FindConflicts <in_cover> <out_cover> <conflict_distance>`



- This tool will help you locate where buildings are within the specified distance; that is, they are in spatial conflict. A buffer will be created around each building or group of connected buildings. The points where the buffers overlap indicate a conflict. An item, FREQUENCY, will be added to the out_cover.PAT, carrying the number of buffers that share each polygon. A FREQUENCY value of 1 means no conflict; a value of 2 or more, according to how many buffers overlap, indicates a conflict area. Buildings connected in one group are not considered conflicting with each other. Only the outer boundary of such a group will be checked with neighboring buildings or groups of buildings.
- The output coverage is created only if conflicts are identified. Since the input buildings are regions, the buffers in the output coverage are also regions with a subclass BUF. You can select and view the conflict areas (the polygons with a FREQUENCY value of 2 or more) and make necessary edits.

Simplify Building: simplifies the boundary or footprint of building polygons while maintaining their essential shape and size.

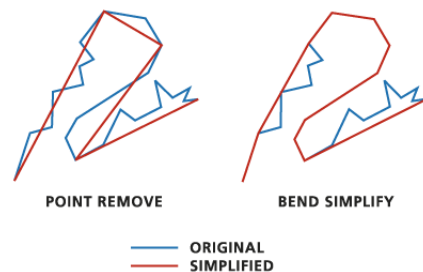
```
SimplifyBuilding <in_cover> <out_cover> <simplification_tolerance> {minimum_area}
{selection_file} {NOT_CHECK | CHECK_CONFLICT}
```



- The input coverage must have a polygon topology.
- Due to the possibility of creating overlapping boundaries, preliminary regions are used as the resulting features. To create fully-built regions from the preliminary regions, use Clean with the POLY option on the out_cover.
- If a selection file is not specified or if it contains no polygons, all polygons in the input coverage are selected for simplification. If the selection file does not contain the polygon feature class or if it does not match the input coverage (that is, the selection file was not derived from the input coverage), the program will stop.

Simplify Line Or Polygon: removes small fluctuations or extraneous bends from a line or polygon, while preserving its essential shape.

```
SimplifyLineOrPolygon <in_cover> <out_cover> <simplification_tolerance> {POINT_REMOVE |
BEND_SIMPLIFY} {NO_ERROR_CHECK | ERROR_CHECK}
```



- If the input coverage already contains intersecting lines or if you want a quick result and don't care about topological errors in the output coverage, use the default option, which is not to Check for topological errors. Any topological errors introduced by the process will not be checked and corrected. If the input coverage contains intersecting lines and you choose to Check for topological errors, it will fail at the input data validation and the program will terminate with a message: "Intersecting lines are found in in_cover. The program is terminated."
- If the input coverage contains no intersecting lines, check the Check for topological errors option to find and avoid errors generated by the simplification process. If any topological errors are found, the involved arcs will be regeneralized using a reduced tolerance. The result will be checked for topological errors again. The process iterates until no more errors are found. With this option, the program will run much longer than with the default option.



Indexes toolset

Contains tools to add or remove attribute indexes.

Drop Index: drops an attribute index from the specified item and INFO table.

`DropIndex <in_info_table> {index_item;index_item...}`

- If there are no indexes on a coverage, the dialog box will not show any fields on which to drop an index.

Index Item: creates an attribute index to increase access speed to the specified item during query operations.

`IndexItem <in_info_table> <index_item>`

- Indexed items speed up selection operations of large INFO files.
- Item indexes are preserved when the coverage or INFO table is copied to a new location.



Items toolset

Contains tools to add or remove items (fields) in INFO tables.

Add Item: adds a blank or zero item to an INFO table.

`AddItem <in_info_table> <out_info_table> <item_name> <item_width> <output_width> <BINARY | CHARACTER | DATE | FLOATING | INTEGER | NUMERIC> {decimal_places} {start_item}`

- Do not insert items before the cover-ID in a feature attribute table.
- Do not insert items before the COUNT item in a grid VAT.
- If item type defines a character, blanks are inserted for each record. If item type defines a numeric item, then zeroes are inserted for each record.

Drop Item: deletes one or more items from an INFO table.

`DropItem <in_info_table> <out_info_table> <drop_item;drop_item...>`

- The output info table can be the same name as the input info table. However, if the output info table already exists, it will be replaced.
- Do not drop items before the User-ID in feature attribute tables. Redefined items will be dropped if their item definition relates to an item that was dropped.

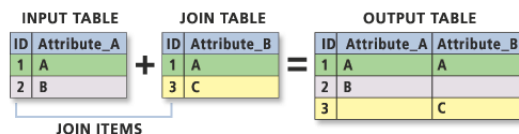


Joins toolset

Contains a tool to join INFO tables.

Join Info Tables: joins the item definitions and values of two tables based on a shared item.

`JoinItem <in_info_table> <join_info_table> <out_info_table> <relate_item> {start_item} {LINEAR | ORDERED | LINK}`



- To maintain the integrity of a feature INFO table, do not insert items before the Input INFO Table-ID (when the output INFO table equals the input INFO table).
- If the same item name is encountered in both tables, the item from the input INFO table is maintained and the join INFO table item is excluded.

- The speed of execution will depend on the organization of the files being joined. In general, LINK is the fastest matching operation, then LINEAR with an indexed relate item, then ORDERED. Although the fastest option, LINK cannot be applied to most cases.



Projections toolset

Contains tools to set a projection or reproject or transform a coverage.

Define Projection: creates or modifies the coordinate system information (including projection parameters, such as datum and spheroid) stored in the coverage's projection definition file (.prj).

`DefineProjection <in_cover> <projection_file>`

- This command can be used if the input dataset or feature class does not have a projection defined. If the input dataset or feature class already has a projection defined, a warning will be raised but the tool will execute successfully.
- Define Projection will not change the coordinates of the output dataset. To project a dataset from one projection to another, you must use the Project command.

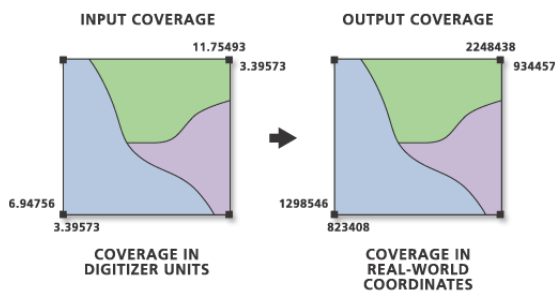
Project: changes the coordinate system of a coverage including its datum or spheroid.

`Project <in_cover> <out_cover> <projection_file>`

- It can convert a dataset from a spherical coordinate system with angular units (such as geographic) to a planar coordinate system with linear units. Most Coverage tools, among them Build and Clean, assume you have a planar, two-dimensional dataset. So if your dataset is in a geographic coordinate system in decimal degrees (DD, angular units), the Project tool projects your dataset to any suitable projected coordinate system in linear units (meters or feet).
- Output projection information can be specified using a projection file or from an empty output coverage. The projection file must contain both input and output projection definitions. Use of a projection file will override any projection information stored in the data's PRJ file.

Transform: moves all features in the coverage based on a set of from and to control points.

`Transform <in_cover> <out_cover> {AFFINE | PROJECTIVE | SIMILARITY}`



- The output coverage must already exist with a tic file containing x,y coordinates for at least two in the desired location and units. Except for the tics, any existing features in the output coverage will be replaced by features from the input coverage.
- The input coverage tic file and the output coverage tic file must contain at least two that have the same Tic-IDs and represent corresponding locations in both coverages. The two coverages do not have to have identical tics; only those tics whose IDs are common to both coverages will be used in the transformation.

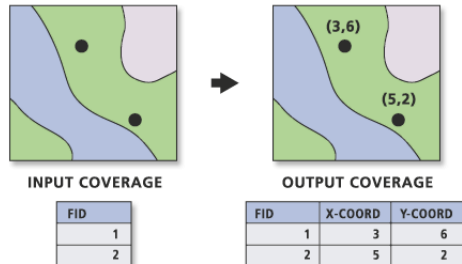


Tables toolset

Contains tools used for editing the associated attribute tables.

Add XY Coordinates: calculates and adds x,y coordinates of labels or points to the coverage PAT or x,y coordinates of nodes to the coverage NAT.

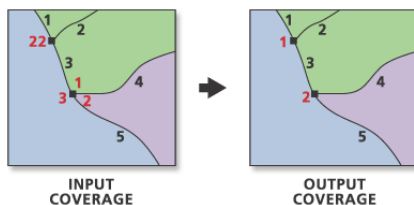
`AddXY <in_cover> {POINT | NODE}`



- If the items X-COORD and Y-COORD already exist, they will be overwritten.
- If the point or node locations are moved after using Add XY Coordinates, the X-COORD and Y-COORD values will not represent the new locations. To update their values to the new location, rerun the tool. The values for X-COORD and Y-COORD are not modified by other tools, such as Project and Transform.
- If your input coverage is in a geographic coordinate system, the X-COORD and Y-COORD represents the longitude and latitude, respectively.

Renumber Nodes: updates arc-node topology by renumbering nodes for coverage arcs and identifies arcs that share the same node locations.

`Renode <in_cover> {from_item} {to_item}`



- If the input coverage has a node attribute table (NAT), Renumber Nodes does the same thing as Build with the NODE option.
- All nodes in the input coverage are sequentially renumbered starting with 1.
- All feature attribute tables as well as polygon topology and arc-node topology are maintained by Renumber Nodes.

Update IDs: updates User-IDs in a coverage after they have been modified in a feature attribute table.

`IDedit <in_cover> <POLY | LINE | POINT | ANNO.subclass>`

- Tools, such as Add Item and Calculate Field, can be used to add or modify User-IDs in a coverage's feature attribute table before Update IDs is used.
- If the Create Labels tool has been used to create new label points for coverage polygons, the polygon User-IDs stored in the coverage PAT are not equal to the new label point User-IDs. Create Labels stores the new label points and their User-IDs in the LAB file. Update IDs may be used to change the label point User-IDs to be equal to the User-IDs stored in the PAT.



Tolerances toolset

Contains a tool to adjust coverage-associated tolerances.

Tolerance: sets the tolerances associated with a coverage.

Tolerance <in_cover> {FUZZY | DANGLE | TIC_MATCH | EDIT | NODESNAP | WEED | GRAIN | SNAP}
{tolerance_value}

- A tolerance value of zero will not be accepted for the following options: FUZZY, EDIT, NODESNAP, WEED, GRAIN, and SNAP.
- If no tolerance type is specified, the default type is FUZZY.



Topology toolset

Contains tools used to develop the topologic relationship within a coverage.

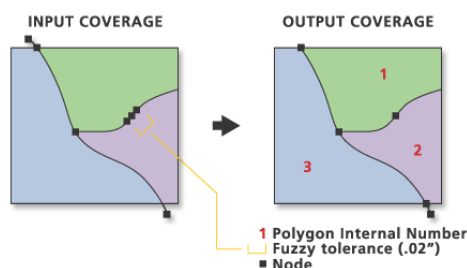
Build: creates or updates feature attribute tables and polygon topology.

Build <in_cover> <POINT | LINE | POLY | NODE | ANNO> {anno_subclass}

- If a coverage feature attribute table exists, the additional items in the feature attribute table will be updated using the old internal number of each of the features specified as the relate item.
- User-defined items in existing feature attribute tables are always maintained.
- When using Build with the POLY option, polygons must have label points to retain their attributes. If there are no attributes, label points are not required to generate a PAT. Polygons containing no label points will be assigned a User-ID of zero. Build does not create polygon labels.
- Build and Clean are similar commands, since they are both used to define coverage topology. The basic difference is that Clean can detect and create intersections but Build cannot. However, since Build does not use a fuzzy tolerance, the coordinates will not be adjusted while topology is being built.

Clean: generates a coverage with correct polygon or arc-node topology.

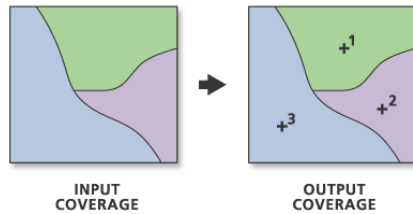
Clean <in_cover> {out_cover} {dangle_length} {fuzzy_tolerance} {POLY | LINE}



- If the input coverage has either PAT or AAT feature attribute tables, they are automatically updated in the output coverage for the POLY option. Only the AAT will be updated when using the LINE option. The internal number of each input coverage feature is used to relate attribute information from the input coverage feature attribute table to the output coverage to ensure that the attributes are properly joined to the output feature attribute tables. Feature User-IDs do not have to be unique to ensure that each input feature keeps its attributes in the output coverage.
- Clean with the POLY option creates one additional polygon called the background or universe polygon. It is always given polygon internal number 1, and its area is the total sum of the areas of all other polygons in the coverage. It is shown as a negative AREA in the PAT.
- The dangle length and fuzzy tolerance for the output coverage are set and verified by Clean.

Create Labels: creates label points for polygons that have no labels and assigns each a User-ID.

`CreateLabels <in_cover> {id_base}`



- The Input Coverage must contain polygon topology.
- After Create Labels, the polygon User-IDs stored in the input coverage PAT are not equal to the new label point User-IDs generated by Create Labels. You must use Build or Update IDs to make them equal.
- If a coverage contains polygons and only some of the polygons have label points, Create Labels will only generate labels in those polygons for which no labels exist if you specify an ID base.

VPF Tile Topology: creates cross-tile topology for all tiled coverages in a Vector Product Format (VPF) database library or creates topology for an individual tile in a VPF library.

`VPFTile <VPF_library> {sig_digits} {93 | 96} {ALL | VPF_cover}`

- Military Standard MIL-STD-2407 (June 28, 1996) refines the definition of cross-tile topology. The VPF Tile Topology command has been updated to meet the new specification. You can use the optional parameter to choose the 93 or 96 VPF Standard.
- VPF Tile Topology works on all tiled coverages of a VPF library or a single coverage within that library. The last optional parameter, VPF_cover, allows you to select a particular coverage in which cross-tile topology should be populated. It is more efficient, however, to implement VPF Tile Topology after all coverages for a library have been converted from ARC/INFO to VPF format.



Workspace Management toolset

Contains a tool to manage coverages within a workspace.

Create Coverage: creates a new, empty coverage.

`Create <out_cover> {template_cover}`

- The coordinate precision of the output coverage is determined by the precision for derived coverages environment, whether or not the template coverage is specified.
- To establish the location of the TICS in the output coverage, specify a template coverage or edit the output coverage manually. You can then use the output coverage as the destination (output) coverage of the Transform tool.

Data Management toolbox

Contains the tools to develop, manage, and manipulate feature classes, datasets, and layers.

Database toolset

Contains tools that improve database performance.

❖ **Compact:** reduces the size and optimizes the performance of a personal geodatabase.

`Compact <in_workspace>`

- You should compact your databases on a regular basis if you do a lot of data entry and deletion.
- Access personal geodatabases are stored in an MDB file. As you use your personal geodatabase, the MDB file may become fragmented on your disk, resulting in a decrease in performance.
- It is recommended that you compact your database when its size increases to more than 250 MB.

◆ **Compress:** removes all unreferenced database states and redundant rows in a version.

`Compress <in_workspace>`

- When you delete a record from a database, it is only marked as deleted; it is not actually removed from the associated table. Therefore, the table will remain the same size after you delete records. To actually remove deleted records from the database, you must compress the database.
- To improve database performance, the database should be compressed periodically.
- Once a database is compressed, deleted records cannot be recovered.
- Only the SDE administrator can perform compression.
- After compressing the database or editing the data, the Analyze command should be executed to update the database statistics for each dataset or feature class. This will improve display and query performance.

Disconnected Editing toolset

Contains tools to check out or update data from a shared data system.

◆ **Check In:** checks datasets back in to the parent ArcSDE geodatabase from a checked out ArcSDE or personal geodatabase.

`CheckIn <in_workspace> <dest_workspace> {NON_RECONCILE | RECONCILE}`

- You must have permission to edit the data you are checking in.
- Once checked in, the changes (edits) will be reflected in the master geodatabase and viewable by all users.

◆ **Check In From Delta:** checks in a geodatabase from a delta geodatabase or an XML file. A delta geodatabase contains only the changes exported from a checkout geodatabase.

`CheckInDelta <in_delta_database> <dest_workspace> {NON_RECONCILE | RECONCILE}`

- Instead of checking in edits directly from the checkout geodatabase, you can export the changes only from the checkout geodatabase to a delta database or delta XML file. Delta databases and XML files are smaller than the original checkout geodatabase.

◆ **Check Out:** checks out data from an ArcSDE geodatabase to an ArcSDE or personal geodatabase for offline editing.

`CheckOut <in_data; in_data...> <out_workspace> <out_name> <DATA | SCHEMA_ONLY> <NO_REUSE | REUSE> <RELATED | NO_RELATED>`

- The tool accepts layers or tables that reference data from one ArcSDE server. Either add them to the list in the dialog box or create a semicolon-delimited list at the command line or in a script.
- The layers and tables must reference versioned ArcSDE feature classes and tables for which you have permissions to edit.
- If the checkout to workspace is a personal geodatabase that does not exist, one will be created.

◆ **Export To Delta:** exports changes in a checkout geodatabase to a delta database or XML file. A delta geodatabase contains changes exported from a checkout geodatabase.

`ExportToDelta <in_workspace> <dest_delta_database>`

- Instead of checking in edits directly from the checkout geodatabase, you can export the changes only from the checkout geodatabase to a delta database or delta XML file. Delta databases and XML files are smaller than the original checkout geodatabase.
- The changes in the delta database or XML file may be checked in as a pull check-in from the master geodatabase.



Domains toolset

Contains tools for the management of domains, both coded and attribute, within a workspace.

❖ **Add Coded Value to Domain:** adds a new value to a domain's coded value list.

`AddCodedValueToDomain <in_workspace> <domain_name> <code> <code_description>`

- A coded value domain can apply to any type of attribute—text, numeric, date, and so on—and specifies a valid set of values for an attribute. For example, a coded value list for a text attribute might include valid pipe material values: CL—cast iron pipe; DL - ductile iron pipe; ACP—asbestos concrete pipe or a coded value list might include the numeric values representing valid pipe diameters: .75—3/4"; 2—2"; 24—24"; and 30—30".

❖ **Assign Domain To Field:** sets the domain for a particular field and optionally for a subtype.

`AssignDomainToField <in_table> <field_name> <domain_name> {subtype_code; subtype_code...}`

- When an attribute domain is associated with a table or feature class, an attribute validation rule is created in the database. This attribute validation rule describes and constrains the valid values of a field type.
- One attribute domain can be associated with multiple fields in the same table, feature class, or subtype as well as in multiple tables and feature classes.

❖ **Create Domain:** creates an attribute domain in the specified workspace.

`CreateDomain <in_workspace> <domain_name> <domain_description> <SHORT | LONG | FLOAT | DOUBLE | TEXT | DATE> {CODED | RANGE} {DEFAULT | DUPLICATE | GEOMETRY_RATIO} {DEFAULT | SUM_VALUES | AREA_WEIGHTED}`

- Coded value domains support only default value and duplicate split policies and default value merge policies.
- Range domains support all split and merge policies. After a split or merge operation, the attribute values of output features are calculated based on the numeric values of the input features and the specified split or merge policy.

❖ **Delete Coded Value From Domain:** removes a value from a coded value domain.

`DeleteCodedValueFromDomain <in_workspace> <domain_name> <code; code...>`

- The Code Value parameter Add Value button is used only in ModelBuilder. In ModelBuilder, where the preceding tool has not been run, or its derived data does not exist, the Code Value parameter may not be populated with values. The Add Value button allows you to add expected values so you can complete the Delete Coded Value From Domain dialog box and continue to build your model.

❖ **Delete Domain:** deletes a domain from a workspace.

`DeleteDomain <in_workspace> <domain_name>`

- A domain cannot be deleted if it is associated with a feature class or table. Use the Remove Domain From Field tool to remove the association between a feature class or table and a domain.

❖ **Domain To Table:** creates a table from an attribute domain.

`DomainToTable <in_workspace> <domain_name> <out_table> <code_field> <description_field> {configuration_keyword}`

- Creating a table from an attribute domain allows for additional editing of the table in ArcMap™. For example, a table could be created from a coded value domain, additional code values could be added to the coded value list, and the Table To Domain tool could be used to update the original domain.

❖ **Remove Domain From Field:** removes an attribute domain association from a feature class field.

`RemoveDomainFromField <in_table> <field_name> {subtype_code; subtype_code...}`

- The Remove Domain From Field function is the opposite operation from the Assign Domain To Field function. Removing a domain from a field removes the association between a field and an attribute domain.
- When a domain is removed from a field, the attribute validation rule for that field is removed from the database.

❖ **Set Value For Range Domain:** sets the minimum and maximum values for an existing range domain.

`SetValueForRangeDomain <in_workspace> <domain_name> <min_value> <max_value>`

- A range domain specifies a valid range of values for a numeric attribute. For example, a valid range of water main pressure values might be between 50 and 75 psi.

❖ **Table To Domain:** creates or updates a coded value domain from a table.

`TableToDomain <in_table> <code_field> <description_field> <in_workspace> <domain_name> <domain_description> <APPEND | REPLACE>`



Feature Class toolset

Contains tools for basic feature class management including appending and integrating multiple feature classes.

❖ **Append Annotation Feature Classes:** combines annotation from multiple input feature classes to create a new annotation feature class.

`AppendAnnotation <input_features; input_features...> <output_feature_class> <reference_scale> {CREATE_CLASSES | ONE_CLASS_ONLY} {NO_SYMBOL_REQUIRED | REQUIRE_SYMBOL} {AUTO_CREATE | NO_AUTO_CREATE} {AUTO_UPDATE | NO_AUTO_UPDATE}`

- The output geodatabase annotation feature class cannot be registered as versioned.
- If you select an output annotation feature class that already exists, the features will be appended to that feature class and the tool will project the annotation features to the destination spatial reference.

❖ **Calculate Default Cluster Tolerance:** calculates a default cluster tolerance value.

`CalculateDefaultClusterTolerance <in_features>`

- The value returned by Calculate Default Cluster Tolerance is the same cluster tolerance value used as a default for other geoprocessing tools.

❖ **Calculate Default Spatial Grid Index:** calculates a spatial grid value used to quickly locate features in a dataset that match the criteria of a spatial search.

`CalculateDefaultGridIndex <in_features>`

- The spatial index calculated by Calculate Default Spatial Grid Index cannot be added to a personal geodatabase because it needs the existing index to maintain its integrity. When you create a new feature class in a personal geodatabase, you can specify a spatial index.
- Spatial grid indexes can be added to ArcSDE geodatabase feature classes using Add Spatial Index.

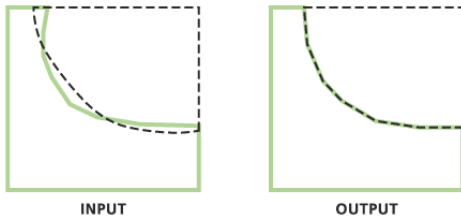
❖ **Create Feature Class:** creates a new feature class.

`CreateFeatureClass <out_path> <out_name> <POLYGON | POINT | MULTIPOINT | POLYLINE>
{template;template...} {DISABLED | SAME_AS_TEMPLATE | ENABLED} {DISABLED |
SAME_AS_TEMPLATE | ENABLED} {spatial_reference} {configuration_keyword}
{spatial_grid_1} {spatial_grid_2} {spatial_grid_3}`

- The type of feature class created depends on the format of the path name specified: a geodatabase feature class for ArcSDE and personal geodatabase path names or a shapefile feature class for a file system folder.
- The Create Feature Class function only creates simple feature classes. Custom feature classes (annotation, dimensions, and so on) can be created in ArcCatalog™.

❖ **Integrate:** compares one or more feature classes and makes any lines, points, or vertices within a specified distance range identical or coincident.

`Integrate <features{Ranks};features{Ranks}...> {cluster_tolerance}`



- The value for cluster tolerance is critical for Integrate. A cluster tolerance that is too large may collapse polygons or merge arcs and points that should not be merged. To minimize error, the value you choose for cluster tolerance should be as small as possible.
- The input features may include any combination of point, multipoint, line, and polygon feature classes.
- Integrate can use feature classes from read-only data, such as CAD or coverages as input. The data will be used as part of the integration process but will not be modified. This can be useful for integrating (snapping) the features in a shapefile, or geodatabase feature class, to the features in the read-only format. To be effective, you may want to assign a higher rank (1) to the read-only format. Copy your inputs before attempting this.

❖ **Update Annotation Feature Class:** updates the input annotation feature class with text attribute fields and optionally populates the value of each new field for every feature in the feature class.

`UpdateAnnotation <in_features> {POPULATE | DO_NOT_POPULATE}`

- This tool will update the schema of the feature class and, optionally, each annotation feature within the feature class. The schema update will add fields to the feature class (bold, italic, text, and so on) and also ensure that there is a symbol within the symbol collection. Without a symbol in the symbol collection, you can't use the improvements for constructing annotation features.

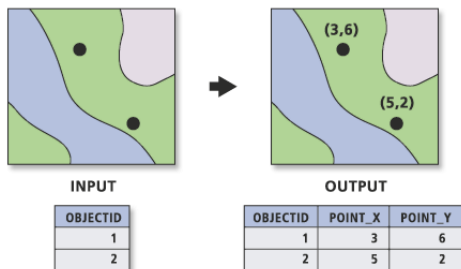


Features toolset

Contains tools to manage and enrich feature classes, such as inspecting and correcting potential errors with data and creating different geometries.

- ❖ **Add XY Coordinates:** adds the field POINT_X and POINT_Y for labels, points, tics, or nodes to the input feature class.

AddXY <in_features>



- If the fields POINT_X and POINT_Y already exist, they will be overwritten.
- If the features are moved after using Add XY Coordinates, the POINT_X and POINT_Y values will not represent the new locations. To update their values to the new location, rerun the tool. The values for POINT_X and POINT_Y are not modified by other tools, such as Project.
- Add XY Coordinates will also add POINT_Z and POINT_M fields when the input features are z and menabled.

- ❖ **Check Geometry:** checks the validity of the geometries of features.

CheckGeometry <in_features>;in_features...> <out_table>

- The output table will have one record for each problem found. If no problems are found, the output table will have no records.
- The output table has the following fields:
 - CLASS—The full path to and name of the feature class in which the problem was found.
 - FEATURE_ID—The feature ID (FID) or object ID (OID) for the feature with the geometry problem.
 - PROBLEM—A short description of the problem.
- The PROBLEM field will contain one of the following: Short segment, Null geometry, Incorrect ring ordering, Incorrect segment orientation, Self-intersections, Unclosed rings, or Empty parts.
- For line features, only the short segments and null geometry problems apply.
- This tool doesn't work with points or multipoints.

- ❖ **Copy Features:** copies the selected features to a new feature class.

CopyFeatures <in_features> <out_feature_class> {configuration_keyword} {spatial_grid_1} {spatial_grid_2} {spatial_grid_3}

- This tool can be used for data conversion since it can read many feature formats (any you can add to a map) and write these to shapefile, personal geodatabase, or SDE geodatabase.
- The valid feature types for copying are point, line, polygon, label, node, tic, link, route, section, and region.

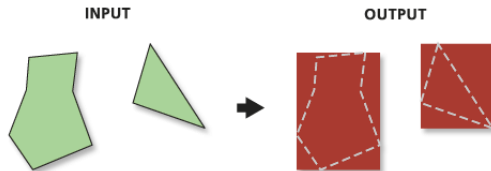
❖ **Delete Features:** deletes features from the input features.

`DeleteFeatures <in_features>`

- Both the geometry and the attributes will be deleted from the affected features.
- ArcSDE or personal geodatabase feature classes, shapefiles, and layers of these data types are valid input features.

Feature Envelope To Polygon: creates polygons from the envelopes of each feature in the input features.

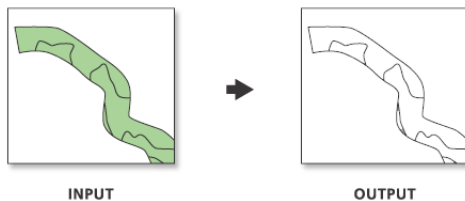
`FeatureEnvelopeToPolygon <in_features> <out_feature_class> {SINGLEPART | MULTIPART}`



- Valid inputs are line, polygon, and annotation.
- When the envelope of a feature is an invalid polygon (height or width is zero), no polygon will be written to the output.
- The multipart parameter will have no effect on the result when the input is a single-part feature.

Feature To Line: creates an output line feature class from input polygon features.

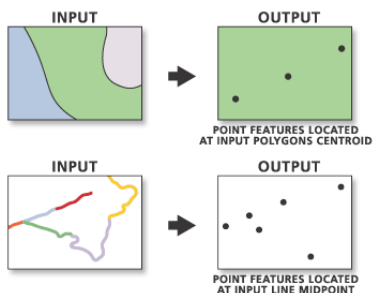
`FeatureToLine <in_features; in_features...> <out_feature_class> {cluster_tolerance} {ATTRIBUTES | NO_ATTRIBUTES}`



- The attributes of the input features are transferred to the output feature class lines.
- To get lines with right and left polygon IDs as attributes from a polygon feature class, use the Polygon To Line tool instead.
- An output line feature will be created for each input line and each input polygon boundary. If input lines or polygon boundaries cross, the output lines will be split into different features at those locations.

Feature To Point: creates a point feature class from the centroids or midpoints of the input features.

`FeatureToPoint <in_features> <out_feature_class> {CENTROID | INSIDE}`

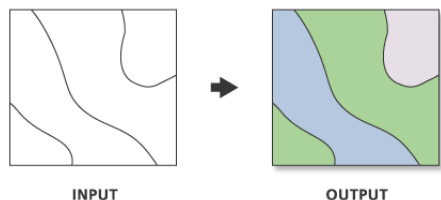


- If the input features are polygons, the centroid option will result in points that are at the center of gravity for that polygon. This may not actually be inside the polygon's area. To ensure that the point created is inside the polygon's area, use the Inside option.

- If the input features are lines and the centroid option is used, the output points will be coincident with the center of the feature's envelope (rectangular window that contains a specific feature). If the inside option is used, the point on the line closest to the center of the feature's envelope will be used.
- If the input features are multipoint and the centroid option is used, the output points will be coincident with the center of the feature's envelope (rectangular window that contains a specific feature). If the Inside option is used, the point (part of the multipoint) closest to the center of the feature's envelope will be used.

Feature To Polygon: creates a new polygon feature class from input line features.

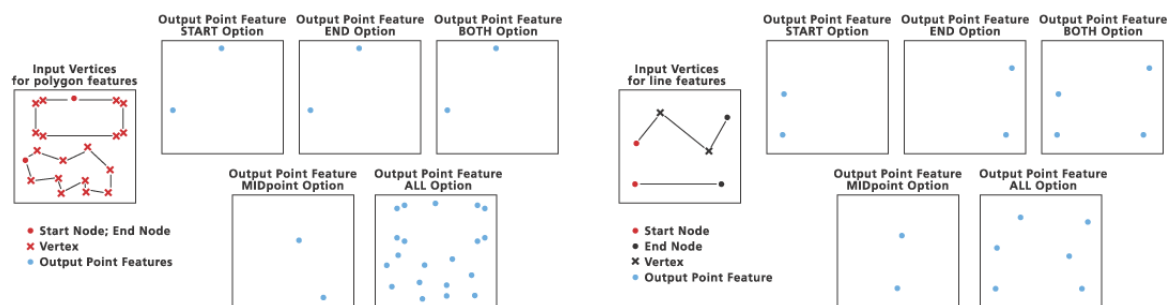
FeatureToPolygon <in_features> <out_feature_class> {cluster_tolerance}
{ATTRIBUTES | NO_ATTRIBUTES} {label_features}



- The input features must be lines or polygons.
- The label features parameter allows for a feature class of label attributes to be applied to the output polygons.

Feature Vertices To Points: creates points from the vertex locations of the input features.

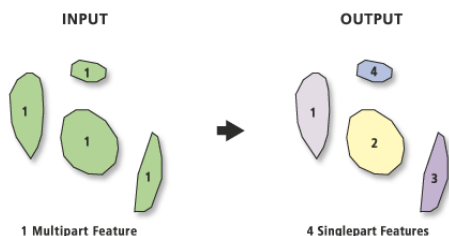
FeatureVerticesToPoints <in_features> <out_feature_class>
{ALL | MID | START | END | BOTHENDS}



- The attributes of the input features are copied to the output feature class points.
- input features can be lines or polygons.

❖ **Multipart To Singlepart:** breaks any multipart features into single features.

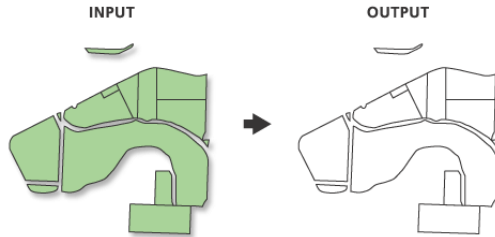
MultipartToSinglepart <in_features> <out_feature_class>



- Features that already have singlepart geometry will not be affected.
- The opposite of this tool is Dissolve, which creates multipart features from single-part features based on a common attribute value.

Polygon To Line: generates a new line feature class from input polygon features.

`PolygonToLine <in_features> <out_feature_class>`



- All input feature classes and/or feature layers must be polygon geometry.
- The right and left polygon FIDs are maintained in the output line feature class.
- Topology is neither created nor maintained with the output of this tool.

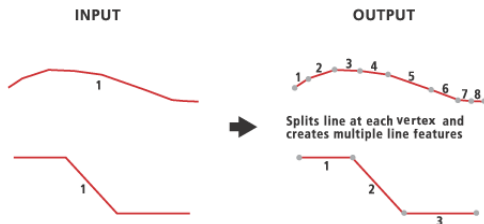
❖ **Repair Geometry:** repairs geometry problems in a feature class or layer.

`RepairGeometry <in_features>`

- The Check Geometry tool can be used to identify feature classes and features within those feature classes that have geometry problems.
- Problems repaired with this tool include null geometry, short segment, incorrect ring ordering, incorrect segment orientation, self-intersections, unclosed rings, and empty parts.
- This tool doesn't work with points or multipoints.

Split Line At Vertices: splits a feature class at any intersection contained in that feature.

`splitLine <in_features> <out_feature_class>`



- All input feature classes and/or feature layers must be line geometry.
- Splits each line feature at every vertex for the newly created feature class.
- The output feature class can be a much larger file, since all the new features have been created. This will depend on how many vertices are in the input feature class.



Fields toolset

Contains tools to add and make changes to the fields in the table of a feature class.

❖ **Add Field:** adds a field to the table of a feature class, layer, or raster catalog.

`AddField <in_table> <field_name> <LONG | TEXT | FLOAT | DOUBLE | SHORT | DATE | BLOB>
{field_precision} {field_scale} {field_length} {field_alias} {NULLABLE | NON_NULLABLE}
{NON_REQUIRED | REQUIRED} {field_domain}`

- Coverages, stand-alone tables, feature classes from SDE and personal geodatabases, layer files, raster catalogs, and shapefiles will work as valid input for this command. VPF and CAD feature data overlays will not work, since they are read-only formats that are not native to ArcGIS.
- The added field will always be displayed at the end of the table.

- ❖ **Assign Default to Field:** creates a default value for a specified field and automatically applies a user-determined value to a certain field for every row added to the table or feature class.

`AssignDefaultToField <in_table> <field_name> <default_value> {subtype_code; subtype_code...}`

- The default value is dependent on the field type chosen in the Field Name parameter. If you pick a field that is type LONG, the default value has to be type LONG too. For example, a valid number would be anything less than 10 numeric characters.
- A field in the feature class or table must be assigned as the subtype field before new subtypes can be added. This is done using the Set Subtype Field tool.

- ❖ **Calculate Field:** calculates the value of a field using an expression.

`CalculateField <in_table> <field> <expression>`

- Calculate Field computes and assigns a value to the specified field of the input table.
- The calculation can only be applied to one field per operation.

- ❖ **Delete Field:** deletes one or more fields from a table or feature class.

`DeleteField <in_table> <drop_field>;drop_field...>`

- Delete Field can be used with any table, SDE or personal geodatabase feature class, coverage, raster catalog, or shapefile.
- You cannot delete fields from nonnative data formats in ArcGIS, such as VPF and CAD datasets, because they are read-only files.

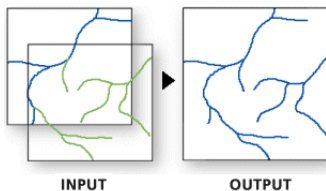


General toolset

Contains tools allowing for some simple dataset changes.

- ❖ **Append:** combines many feature classes into one feature class.

`Append <inputs;inputs...> <target> {TEST | NO_TEST}`



- Append adds the input features to an existing output point, line, or polygon feature class; table; raster dataset; or raster catalog.
- Use Append when you want to combine two or more adjacent layers into one large layer that contains all their features.
- All input features must be of the same feature type (all area features, all line, or all point).

- ❖ **Copy:** copies feature datasets, feature classes, or tables and pastes them to another location.

`Copy <in_data> <out_data> {data_type}`

- The Copy tool copies data from one location and name to a new location and name.
- The input and the output of Copy will be the same data type. Therefore, if the input type is a shapefile, the output type will also be shapefile. Be sure the output database supports the output data type. For example, do not try to copy a shapefile into a geodatabase.

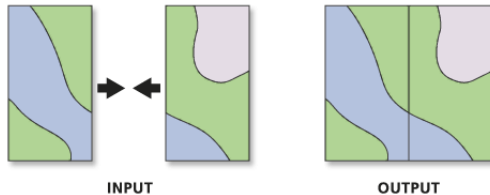
- ❖ **Delete:** deletes feature datasets, feature classes, rasters, or tables.

`Delete <in_data> {data_type}`

- The data to be deleted should not be open anywhere on the operating system; that is, no other user should have the file open. For example, a feature class cannot be deleted if it is open in ArcMap.
- When deleting items with the Delete tool, ancillary files associated with the item are also deleted. For example, the metadata, projection, and index files that may accompany a shapefile will be deleted if the shapefile is deleted.

❖ **Merge:** Combines input features from multiple input sources (of the same data type) into a single, new output feature class.

`Merge <inputs; inputs...> <output> {field_mappings}`



- Use Merge when there are features from multiple input sources that need to be combined into one feature class.
- Input data sources need not be adjacent; overlap is allowed.
- The type of input data, such as polygons or tables, must be the same for all inputs.
- A single output field can be generated from multiple input fields. This happens when more than one input feature class or table contains a field of the same name, or it can happen, if a new field is created and the contents of the output field are generated from multiple (differently named), user-selected fields.
- Merge identifies fields based on their name, not their data type. Input fields are identified by their name and grouped into an output field of the same name.
- The data type of an output field will default to the same as the data type of the first input field (of that name) it encounters. The data type can be changed manually at any time to any valid data type. All valid data types will be listed if the tools dialog box is used.

❖ **Rename:** changes the name of data, such as feature datasets, feature classes, rasters, tables, or toolboxes.

`Rename <in_data> <out_data> {data_type}`

- The output name cannot already exist.
- The data to be renamed should not be open anywhere on the operating system; that is, no other user should be accessing the data. For example, a feature class cannot be renamed if it is open in ArcMap.
- When renaming items with the Rename tool, ancillary files associated with the item are also renamed. For example, the metadata, projection, and index files that may accompany a shapefile will also be renamed.

❖ **Select Data:** selects any data type on disk as input. This tool is intended for use within ModelBuilder, not at the command line.

`SelectData <in_data_element> <out_data_element>`

- The Select Data tool is intended for use only in ModelBuilder.
- Selecting the child from the parent using this tool enables you to continue processing after performing a task where the output data is a container, such as a feature dataset, and the next tool in the model requires a feature class.

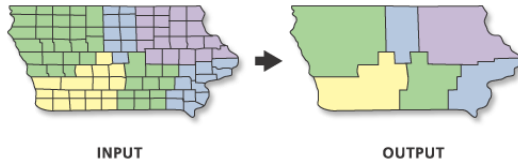


Generalization toolset

Contains tools to derive data with less detail and complexity from a dataset.

❖ **Dissolve:** aggregates features based on one or more specified attributes.

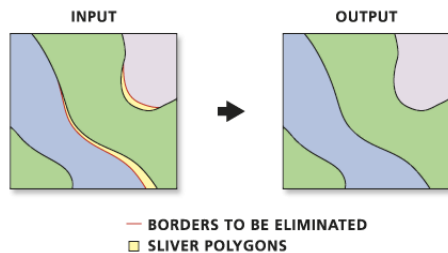
```
Dissolve <in_features> <out_feature_class> {dissolve_field;dissolve_field...}
{field{Statistics_Type}; field{Statistics_Type}...} {MULTI_PART | SINGLE_PART}
```



- The attributes of the aggregated features may be summarized using a statistic type. For example, when aggregating sale territories, the revenue for each feature within a territory could be summed to obtain the total sales revenue for that territory (revenue sum).
- The statistic type used to summarize attributes is added to the output feature class as a single field: `statistic_field`.

Eliminate: eliminates features.

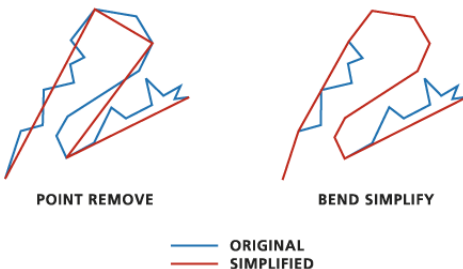
```
Eliminate <in_features> <out_feature_class> {LENGTH | AREA}
```



- Features to be eliminated are determined by a selected feature set applied to a polygon layer. The selected set must be determined in a previous step by using Select Layer By Attribute, using Select Layer By Location, or querying a map layer in ArcMap.
- Only the selected set of polygons from a temporary feature layer will be eliminated.

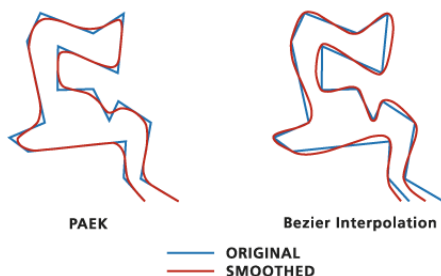
◆ **Simplify Line:** removes small fluctuations or extraneous bends from a line in a feature class.

```
SimplifyLine <in_features> <out_feature_class> <POINT_REMOVE | BEND_SIMPLIFY> <tolerance>
{FLAG_ERRORS | RESOLVE_ERRORS} {KEEP_COLLAPSED_POINTS | NO_KEEP}
```



◆ **Smooth Line:** reduces the number of segments used to represent a line in a feature class.

```
SmoothLine <in_features> <out_feature_class> <PAEK | BEZIER_INTERPOLATION> <tolerance>
{FIXED_CLOSED_ENDPOINT | NO_FIXED}
```



- The Polynomial Approximation with Exponential Kernel (PAEK) algorithm produces smoothed lines; each may have more vertices than its source line. The tolerance you specify is the length of a moving path used in calculating the new vertices. The longer the length, the more smoothed the line. A small tolerance will result in a long processing time, so begin with a relatively large tolerance and a small number of lines.
- The BEZIER_INTERPOLATION algorithm does not require any tolerance. If you are using the command line or scripting, you still need to enter zero in place of the smooth_tolerance.



Indexes toolset

Contains tools to create, alter, and remove indexes.

- ❖ **Add Attribute Index:** adds an index to an existing table, feature class, shapefile, coverage, or attributed relationship class.

```
AddIndex <in_table> <fields;fields...> {index_name} {NON_UNIQUE | UNIQUE} {NON_ASCENDING | ASCENDING}
```

- An attribute index can improve the performance of queries against your data.
- Attribute indexes are used with tables and feature classes, whereas spatial indexes are used with graphical queries of spatial features within feature classes.

- ❖ **Add Spatial Index:** creates a new spatial index for a feature class in an ArcSDE geodatabase.

```
AddSpatialIndex <in_features> {spatial_grid_1} {spatial_grid_2} {spatial_grid_3}
```

- This command will only work with shapefile or feature classes in an SDE geodatabase.
- The spatial index cannot be added to a personal geodatabase because it needs the existing index to maintain its integrity. When you create a new feature class in a personal geodatabase, you can specify a spatial index or accept the default spatial index.
- Indexed items speed up selection and relate operations.
- A load-only mode disables spatial index management until loading is completed.
- An attribute index cannot be added to a versioned feature class.

- ❖ **Remove Attribute Index:** deletes an index from an existing table, feature class, shapefile, coverage, or attributed relationship class.

```
RemoveIndex <in_table> <index_name;index_name...>
```

- Attribute indexes are used with tables and feature classes, whereas spatial indexes are used with graphical queries of spatial features within feature classes.
- Indexed items speed up ArcInfo selection and relate operations.

- ❖ **Remove Spatial Index:** deletes the spatial index for a feature class in an ArcSDE geodatabase.

```
RemoveSpatialIndex <in_features>
```

- The spatial index can only be deleted from an ArcSDE feature class that is not versioned.

- An ArcSDE geodatabase has up to three spatial indexes. They are all subindexes of the parent index, so if you remove the spatial index, all subindexes will be removed.
- The spatial index cannot be deleted from a personal geodatabase, because it needs the index to maintain its integrity.

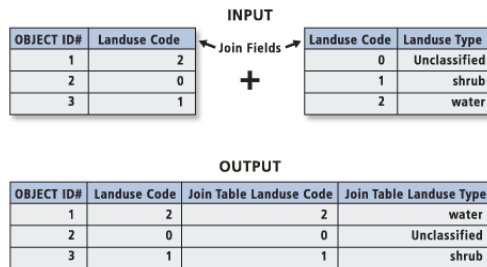


Joins toolset

Contains tools to add or remove a table join.

❖ **Add Join:** links a layer to a table (or a table to a table) based on a common field.

`AddJoin <in_layer_or_view> <in_field> <join_table> <join_field> {KEEP_ALL | KEEP_COMMON}`



- This tool is not limited to ArcMap; it works on layers and table views in ArcCatalog and in scripts. The Make Feature Layer tool makes a layer for a feature class, and the Make Table View creates a table view from an input table or feature class.
- The join table can be any of the following types of tables: a geodatabase table (SDE or personal), a dBASE file, an INFO table, or an OLE DB table.
- Indexing the fields in the input layer or table view and join table on which the join will be based can improve performance. This can be done with the Add Attribute Index tool or by right-clicking the input in ArcCatalog and using the dialog box to add an index to the desired field.

❖ **Remove Join:** removes the link between two tables.

`RemoveJoin <in_layer_or_view> <join_name>`

- The join name is the name of the table that was joined to the input layer or table view.



Layers and Table Views toolset

Contains tools for creating and manipulating layers, layer files, and table views.

❖ **Make Feature Layer:** creates a temporary layer.

`MakeFeatureLayer <in_features> <out_layer> {where_clause} {workspace} {field_info}`

- The feature layer can be used as input to any geoprocessing tool that accepts a feature class as input.
- The temporary feature layer can be saved as a layer file using the Save To Layer File tool or saved as a new feature class using the Copy Features tool.
- If a SQL expression is used but returns nothing, the output feature layer will be empty.

❖ **Make Query Table:** represents the results of a SQL query to a database in a layer or table view.

`MakeQueryTable <in_table; in_table...> <out_table> {USE_KEY_FIELDS | ADD_VIRTUAL_KEY_FIELD | NO_KEY_FIELD} {in_key_field; in_key_field...} <field{Alias}; field{Alias}...> {where_clause}`

- Make Query Table accepts data from an ArcSDE geodatabase, a personal geodatabase, or an OLE DB connection.
- All input feature classes or tables must be from the same input workspace.

- If a shape column is added to the field list, the result is a layer; otherwise, it is a table view.

❖ **Make Raster Catalog Layer:** makes a temporary raster catalog layer.

```
MakeRasterCatalogLayer <in_raster_catalog> <layer_name> {where_clause} {workspace}
{field_info}
```

❖ **Make Raster Layer:** makes a temporary raster layer.

```
MakeRasterLayer <in_raster> <out_raster_layer> {where_clause} {envelope}
```

❖ **Make Table View:** creates a temporary table.

```
MakeTableView <in_table> <out_name> {where_clause} {workspace} {field_info}
```

- This tool is commonly used to create a table view with a selected set of attributes or fields.
- ArcCatalog does not display these table views, but they can be used as inputs to other geoprocessing tools in the current ArcGIS session. Once ArcGIS exits, the tables in memory are removed.
- Table views created in ArcCatalog cannot be used in ArcMap.
- If a SQL expression is used but returns nothing, the output table view will be empty.

❖ **Make XY Event Layer:** creates a temporary event layer with x,y coordinates.

```
MakeXYEventLayer <table> <in_x_field> <in_y_field> <out_layer> {spatial_reference}
```

- This tool allows you to create a layer based on x and y columns from an input table.
- If the table is editable, you will be able to edit the layer. However, you won't be able to interactively move a point on the map; you must change the coordinates in the table.

❖ **Save To Layer File:** creates a layer file on disk.

```
SaveToLayerFile <in_layer> <out_layer>
```

- The input layer can be an in-memory layer created by the Make Layer tool, a layer file stored on disk, or a feature layer in ArcMap.

❖ **Select Layer By Attribute:** creates, updates, or removes the selection on a layer or table view using an attribute query.

```
SelectLayerByAttribute <in_layer_or_view> {NEW_SELECTION | ADD_TO_SELECTION | REMOVE_
FROM_SELECTION | SUBSET_SELECTION | SWITCH_SELECTION | CLEAR_SELECTION} {where_clause}
```

- The input must be a feature layer or a table view. It cannot be a feature class or table.
- This tool is not limited to working in ArcMap; it works on layers and table views in ArcCatalog and in scripts. The Make Feature Layer tool makes a layer for a feature class, and the Make Table View tool does the equivalent for a table.

❖ **Select Layer By Location:** creates, updates, or removes the selection of features in a layer based on a spatial relationship.

```
SelectLayerByLocation <in_layer> {INTERSECT | WITHIN_A_DISTANCE | COMPLETELY_CONTAINS |
COMPLETELY_WITHIN | HAVE_THEIR_CENTER_IN | SHARE_A_LINE_SEGMENT_WITH | BOUNDARY_TOUCHES
| ARE_IDENTICAL_TO | CROSSED_BY_THE_OUTLINE_OF | CONTAINS | CONTAINED_BY} {select_
features} {search_distance} {NEW_SELECTION | ADD_TO_SELECTION | REMOVE_FROM_SELECTION |
SUBSET_SELECTION | SWITCH_SELECTION}
```

- The input must be a feature layer or a table view. It cannot be a feature class or table.
- This tool is not limited to working in ArcMap; it works on layers and table views in ArcCatalog and in scripts. The Make Feature Layer tool makes a layer for a feature class, and the Make Table View tool does the equivalent for a table.

- If an extent or a definition query is present on the input layer or table view, only those features or rows that match the extent and/or definition query will be available to be selected.

Projections and Transformations toolset

Contains tools to set the projection as well as reproject or transform a dataset.


❖ **Define Projection:** sets the projection information for a dataset.

`DefineProjection <in_dataset> <coordinate_system>`

- This command can be used if the input dataset or feature class does not have a projection defined. If the input dataset or feature class has a projection defined, a warning will be raised but the tool will execute successfully.
- The coordinate system information of the input is created or modified by this tool. No separate output feature class will be created.

Feature (Projections and Transformations) toolset

Contains tools to convert a geographic dataset from one coordinate system to another.

 ❖ **Batch Project:** changes the coordinate system of one or more feature classes including the datum or spheroid.

`BatchProject <input_feature_class_or_dataset; input_feature_class_or_dataset...>
<output_workspace> {output_coordinate_system} {template_dataset} {transformation}`

- This tool will not work with layers as input.
- All input feature classes and/or feature datasets are valid inputs to this tool.
- If you have a feature class that does not have a defined projection and PRJ file, then use the Define Project tool.
- The name of the output feature classes will be based on the name of the input feature class. For instance, if the input is c:\myworkspace\Gondor.shp, the output feature class will be named gondor. If the name already exists in the output workspace, a number will be appended to the end to make it unique (for example, _1).

❖ **Create Spatial Reference:** creates spatial reference and domains.

`CreateSpatialReference {spatial_reference} {spatial_reference_template} {xy_domain}
{z_domain} {m_domain} {template;template...} {expand_ratio}`

- Setting the spatial reference sets the coordinate system, spatial domains, and precision. The spatial domains and precision of the output spatial reference can be further modified using XY Domain, Z Domain, M Domain, Template XYDomains, and Grow XYDomain By Percentage.
- Template XYDomains does not have to be in the same coordinate system as that specified in Spatial Reference or Spatial Reference Template. If they are different, the extents will be projected to match.

❖ **Project:** changes the coordinate system of a feature class including its datum or spheroid.

`Project <in_dataset> <out_dataset> <out_coordinate_system> {transform_method;
transform_method...}`

- All input feature classes and/or feature layers are valid inputs to this tool.
- If you have a feature class that does not have a defined projection and PRJ file, then use the Define Projection tool first.



Raster (Projections and Transformations) toolset

Contains tools to set the projection, and to reproject, reorient, or relocate a raster dataset.

- ❖ **Flip**: flips a raster dataset along a horizontal axis.

`Flip <in_raster> <out_raster>`



↓ Flip



- Flip flips the grid from top to bottom along the horizontal axis through the center of the region.

- ❖ **Mirror**: flips a raster dataset along the vertical axis.

`Mirror <in_raster> <out_raster>`



Mirror



- Mirror flips the raster from left to right along the vertical axis through the center of the region.

- ❖ **Project Raster**: converts a raster dataset between two coordinate systems.

`ProjectRaster <in_raster> <out_raster> <out_coordinate_system> {NEAREST | BILINEAR | CUBIC} {cell_size}`

- The coordinate system defines how your raster data is projected.
- The NEAREST option, which performs a nearest neighbor assignment, is the fastest of the three interpolation methods. It is primarily used for categorical data, such as a land use classification, because it will not change the cell values. Do not use NEAREST for continuous data, such as elevation surfaces.
- The BILINEAR option, bilinear interpolation, determines the new value of a cell based on a weighted distance average of surrounding cells. The CUBIC option, cubic convolution, determines the new cell value by fitting a smooth curve through the surrounding points. These are most appropriate for continuous data and may cause some smoothing; also, cubic convolution may result in the output raster containing values outside the range of the input raster. It is not recommended that BILINEAR or CUBIC be used with categorical data because the cell values may be altered.

- ❖ **Rescale**: scales a raster by the specified x and y scale factors.

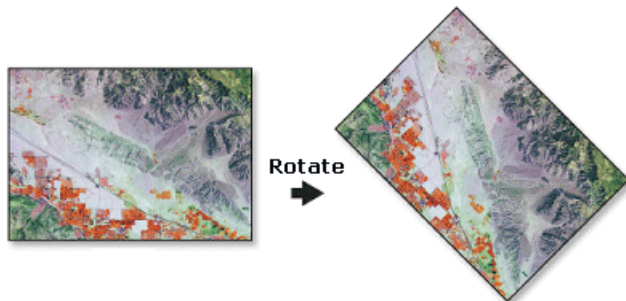
`Rescale <in_raster> <out_raster> <x_scale> <y_scale>`

- The scale factor must be a positive number.
- The scale factor can be set for both the x and y directions.
- A scale factor greater than one means the image will be rescaled to a larger dimension, resulting in a larger extent because of a larger cell size.

- A scale factor less than one means the image will be rescaled to a smaller dimension, resulting in a smaller extent because of a smaller cell size.

❖ **Rotate:** rotates a raster dataset around a specified point by a specified angle.

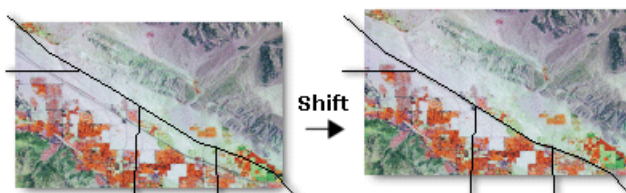
`Rotate <in_raster> <out_raster> <angle> {pivot_point} {NEAREST | BILINEAR | CUBIC}`



- Rotation is, by default, around the lower left corner of the raster. The rotation point can be changed with the optional Pivot Point parameter.
- Resampling is only done if the angle is not a multiple of 90.
- The rotation angle specified must be between 0 and 360 degrees.

❖ **Shift:** shifts a raster by the specified x and y shift values.

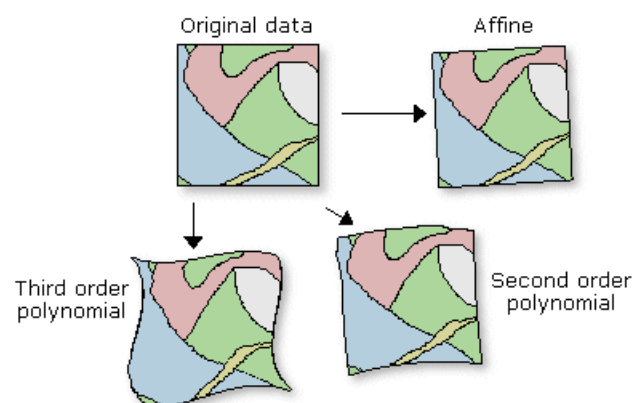
`Shift <in_raster> <out_raster> <x_value> <y_value> {in_snap_raster}`



- The cell size of the output raster will be the same as that of the input raster.
- The number of rows and columns in the output raster will be the same as those of the input raster, no matter what parameters are specified.
- The coordinates of the lower-left corner of the output raster will be offset from the input raster by the x and y shift coordinate values specified.
- Shift does not perform any resampling or warping.

❖ **Warp:** transforms or rubber sheets a raster dataset along a set of links using a polynomial transformation.

`Warp <in_raster> <source_control_points;source_control_points...> <target_control_points;target_control_points...> <out_raster> {POLYORDER1 | POLYORDER2 | POLYORDER3} {NEAREST | BILINEAR | CUBIC}`




- The default polynomial order (1) will perform an affine transformation.
- Warp is useful when the raster requires a systematic geometric correction that can be modeled with a polynomial. A spatial transformation can invert or remove a distortion by using polynomial transformation of the proper order. The higher the order, the more complex the distortion that can be corrected. The higher orders of polynomial will involve progressively more processing time.
- To determine the minimum number of links necessary for a given order of polynomial, use the following formula:

$$n = (p + 1) (p + 2) / 2$$
 where n is the minimum number of links required for a transformation of polynomial order p. It is strongly suggested that you use more than the minimum number of links.


Raster toolset

Contains tools to create and manage raster datasets and raster catalogs.

 **Batch Build Pyramids:** allows you to build pyramids on multiple raster datasets.

`BatchBuildPyramids <input_raster_datasets;input_raster_datasets...>`

- Building pyramids for a raster dataset will improve the display performance of large raster datasets.
- Pyramids can only be built for raster datasets that do not have internal pyramids.
- Pyramids cannot be built for raster catalogs, but they can be built for each raster catalog member.

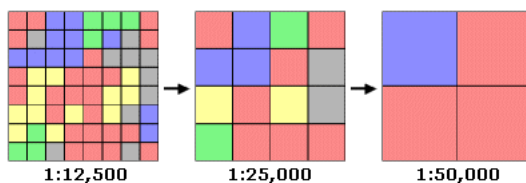
 **Batch Calculate Statistics:** allows you to calculate statistics on multiple raster datasets.

`BatchCalculateStatistics <input_raster_datasets;input_raster_datasets...>`
`{number_of_columns_to_skip} {number_of_rows_to_skip} {ignore_values;ignore_values...}`

- The input raster datasets can be any valid raster dataset that ArcGIS recognizes. This tool can calculate statistics in a batch process.
- Calculating statistics for a raster is required for rendering your raster dataset with any sort of contrast stretch.

 **Build Pyramids:** builds raster pyramids for a raster dataset.

`BuildPyramids <in_raster_dataset>`



- Building pyramids for raster datasets will help improve the display speed.
- You only need to build pyramids once per dataset, then the pyramids can be accessed every time you display that raster dataset.

 **Calculate Statistics:** calculates statistics for a raster dataset.

`calculateStatistics <in_raster_dataset> {x_skip_factor} {y_skip_factor} {ignore_values; ignore_values...}`

- Calculating statistics for a raster is required for rendering your raster dataset with any sort of contrast stretch.
- A skip factor is the parameter that controls the portion of the raster dataset used when calculating the statistics. The input value indicates the horizontal or vertical skip factor, where a factor of one will use each pixel and a value of two will use every second pixel. The skip factor can only range from one to the number of columns/rows.

- The ignore value allows you to exclude a specific value from the calculation of statistics. You may want to ignore a value if it is a NoData value or if it will skew your calculation.

❖ **Clip:** creates a rectangular spatial subset of a raster dataset.

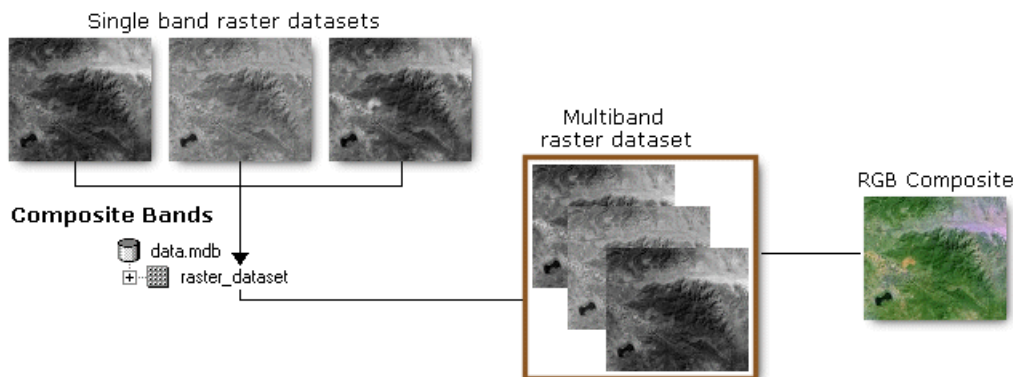
`Clip <in_raster> <rectangle> <out_raster>`



- The Clip tool allows you to extract a portion of a raster dataset based on a rectangular extent.
- The minimum and maximum x and y extents allow you to define the clip extents for your output raster dataset.
- The extent values must be in the same spatial coordinates and units as the raster dataset.

❖ **Composite Bands:** creates a multiband raster dataset from one or more raster datasets.

`CompositeBands <in_rasters;in_rasters...> <out_raster>`



- A multiband raster dataset can be a GRID stack, TIFF, IMG, or SDE raster. With the exception of GRID stacks, none of these formats support the addition or removal of individual raster bands. For this reason, there is no tool to make such changes. The Composite Bands tool can be used to create a new multiband raster dataset that includes only the rasters you want.
- The Raster Analysis Environment Settings are not valid for this tool.

❖ **Copy Raster:** converts a raster dataset to a GRID, TIFF, IMAGINE, or geodatabase raster dataset or loads raster datasets into a raster catalog.

`CopyRaster <in_raster> <out_raster_dataset> {configuration_keyword} {background_value} {nodata_value} {NONE | ONEBITTO8BIT} {NONE | COLORMAPTORGb} {1_BIT | 2_BIT | 4_BIT | 8_BIT_UNSIGNED | 8_BIT_SIGNED | 16_BIT_UNSIGNED | 16_BIT_SIGNED | 32_BIT_UNSIGNED | 32_BIT_SIGNED | 32_BIT_FLOAT | 64_BIT}`

- There is no particular control for specifying which output format to generate. The format is determined by the output raster. If a .tif extension is used, the output raster will be a TIFF. If the extension is .img, an ERDAS IMAGINE file is created. If the name contains an ArcSDE database connection, an ArcSDE raster dataset is created. If the name does not contain ArcSDE database connection information or a file extension, an ArcInfo GRID is generated. If the raster was a multiband raster, a GRID stack will be created.

❖ **Copy Raster Catalog Items:** makes a copy of a raster catalog including its contents.

```
CopyRasterCatalogItems <in_raster_catalog> <out_raster_catalog> {configuration_keyword}
{spatial_grid_1} {spatial_grid_2} {spatial_grid_3}
```

- The input and output of this tool is a geodatabase raster catalog.
- If your raster catalog output is to an ArcSDE geodatabase, a configuration keyword can be set.
- Personal geodatabases must have one spatial index (grid). ArcSDE geodatabases can have up to three spatial indexes.

❖ **Create Raster Catalog:** creates an empty raster catalog in a geodatabase.

```
CreateRasterCatalog <out_path> <out_name> {raster_spatial_reference} {spatial_reference}
{configuration_keyword} {spatial_grid_1} {spatial_grid_2} {spatial_grid_3} {MANAGED |
UNMANAGED}
```

- Raster datasets within raster catalogs in a geodatabase can be managed in two ways: managed or not managed by the geodatabase. Having the raster catalog managed by the geodatabase means that the raster datasets will be stored within the personal geodatabase. When a row is deleted from the catalog, it is deleted from the geodatabase. When you do not have your raster managed by the geodatabase, there will only be a pointer connecting the raster catalog row to the raster dataset.
- Raster catalogs stored in ArcSDE are always managed.
- When creating a raster catalog in an ArcSDE geodatabase, the raster dataset name cannot have spaces. You can use underscores to separate words.

❖ **Create Raster Dataset:** creates an empty raster dataset in a geodatabase.

```
CreateRasterDataset <out_path> <out_name> {cellsize} {8_BIT_UNSIGNED | 1_BIT | 2_BIT |
4_BIT | 8_BIT_SIGNED | 16_BIT_UNSIGNED | 16_BIT_SIGNED | 32_BIT_UNSIGNED | 32_BIT_SIGNED
| 32_BIT_FLOAT | 64_BIT} {raster_spatial_reference} {number_of_bands} {configuration_
keyword} {pyramids} {tile_size} {compression} {pyramid_origin}
```

- Specify the X/Y Domain when defining the spatial reference; if you don't, your datasets may not be visible in the display.
- It is best to choose to build pyramids, because pyramids can speed up the display of raster data since the server or computer returns only the data at a specified resolution that is required for the display.
- The raster dataset can be saved in a geodatabase or on the file system. If you are saving it to the file system, specify the file type extension (.tif for a TIFF format or .img for an ERDAS IMAGINE format). No extension is needed to create a GRID.

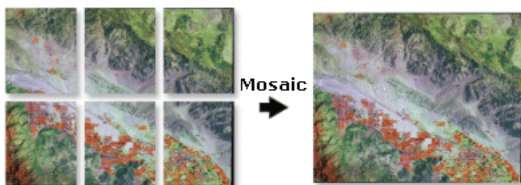
❖ **Delete Raster Catalog Items:** deletes the raster datasets contained in a raster catalog.

```
DeleteRasterCatalogItems <in_raster_catalog>
```

- Only geodatabase raster catalogs are valid inputs.

❖ **Mosaic:** mosaics multiple rasters into a single raster.

```
Mosaic <inputs;inputs...> <target> {LAST | FIRST | BLEND | MEAN | MINIMUM | MAXIMUM}
{REJECT | FIRST | LAST | MATCH} {background_value} {nodata_value} {NONE | ONEBITTO8BIT}
{mosaicking_tolerance}
```




- Mosaic is useful when a set of adjacent rasters needs to be merged into one entity and also when minimizing the abrupt changes along the boundaries of the overlapping rasters.

- For mosaicking of discrete data, the First, Minimum, or Maximum options will give the most meaningful results. The Blend and Mean options are best suited for continuous data.
- For floating-point input rasters of different resolutions, it is recommended to resample all the data using bilinear interpolation or cubic convolution before running Mosaic. Otherwise, Mosaic will automatically resample the rasters using nearest neighbor (which is not appropriate for the continuous type of data) and the current cell size setting in the analysis environment and before mosaicking.

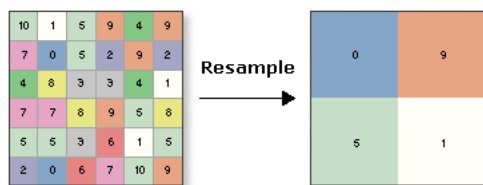
 **Mosaic To New Raster:** mosaics multiple rasters into a new, single raster dataset.

`MosaicToNewRaster <input_rasters> <input_rasters...> <output_location> <raster_dataset_name_with_extension> {coordinate_system_for_the_raster} {8_BIT_UNSIGNED | 1_BIT | 2_BIT | 4_BIT | 8_BIT_SIGNED | 16_BIT_UNSIGNED | 16_BIT_SIGNED | 32_BIT_UNSIGNED | 32_BIT_SIGNED | 32_BIT_FLOAT | 64_BIT} {cell_size} <number_of_bands> {FIRST | LAST | BLEND | MEAN | MINIMUM | MAXIMUM} {REJECT | FIRST | LAST | MATCH}`

- The input rasters are all the raster datasets you want to mosaic together. The inputs must have the same number of bands; otherwise, the Mosaic tool used in the model will not run.
- If your raster dataset is stored in the file system, you must specify the proper extension for the new raster dataset; if it is to be stored as a GRID, do not add an extension. Valid outputs to a file system include ESRI GRID (no extension), ERDAS IMAGE (.img), and TIFF (.tif).

 **Resample:** changes the cell size of a grid.

`Resample <in_raster> <out_raster> <cell_size> {NEAREST | BILINEAR | CUBIC}`



- The lower-left corner of the output raster will be the same map space coordinate location as the lower-left corner of the input raster.
- The numbers of rows and columns in the output raster are determined as follows:

$$\text{columns} = (\text{xmax} - \text{xmin}) / \text{cell size}$$

$$\text{rows} = (\text{ymax} - \text{ymin}) / \text{cell size}$$
 If there is any remainder from the above equations, then rounding of the number of columns and/or rows is performed.
- The current environment settings of cell size (if one is specified) and extent are applied to the output. The mask is not used.
- The Nearest option, which performs a nearest neighbor assignment, is the fastest of the interpolation methods. It is used primarily for categorical data, such as a land use classification, since it will not change the values of the cells. The maximum spatial error will be one-half the {cellsize}.
- The Bilinear option, which performs a bilinear interpolation, determines the new value of a cell based on a weighted distance average of the four nearest input cell centers. It is useful for continuous data and will cause some smoothing of the data.
- The Cubic option, which performs a cubic convolution, determines the new value of a cell based on fitting a smooth curve through the 16 nearest input cell centers. It is appropriate for continuous data, although it may result in the output raster containing values outside the range of the input raster. It is geometrically less distorted than the raster achieved by running the nearest neighbor resampling algorithm. The disadvantage of the Cubic option is that it requires more processing time. In some cases, it can result in output cell values outside the range of input cell values. If this is unacceptable, use Bilinear instead.



Relationship Classes toolset

Contains tools to create associations between feature classes as well as feature classes and tables.

- ◆ **Create Relationship Class:** creates a relationship class to store an association between features in two feature classes or tables.

```
CreateRelationshipClass <origin_table> <destination_table> <out_relationship_class>
<SIMPLE | COMPOSITE> <forward_label> <backward_label> <NONE | FORWARD | BACKWARD | BOTH>
<ONE_TO_ONE | ONE_TO_MANY | MANY_TO_MANY> <NONE | ATTRIBUTED> <origin_primary_key>
<origin_foreign_key> {<destination_primary_key> {<destination_foreign_key>
```

- Relationships can exist between spatial objects (features in feature classes), nonspatial objects (rows in a table), or spatial and nonspatial objects.
- Once created, a relationship class cannot be modified; you can only add, delete, or refine its rules. Relationship classes can be deleted and renamed using ArcCatalog in the same manner as any other object in the database.
- Relationship classes are used to pass messages between related fields or features to notify each other of changes.

- ◆ **Table To Relationship Class:** creates an attributed relationship class from the origin, destination and relationship tables.

```
TableToRelationshipClass <origin_table> <destination_table> <out_relationship_class>
<SIMPLE | COMPOSITE> <forward_label> <backward_label> <NONE | FORWARD | BACKWARD | BOTH>
<ONE_TO_ONE | ONE_TO_MANY | MANY_TO_MANY> <relationship_table> <attribute_
fields;attribute_fields...> <origin_primary_key> <origin_foreign_key> <destination_
primary_key> <destination_foreign_key>
```

- Relationships can exist between spatial objects (features in feature classes), nonspatial objects (rows in a table), or spatial and nonspatial objects.
- Once created, a relationship class cannot be modified; you can only add, delete, or refine its rules. Relationship classes can be deleted and renamed using ArcCatalog in the same manner as any other object in the database.
- Table To Relationship Class creates a table in the database containing the selected attribute fields of the relationship table. These fields are used to store attributes of the relationship itself that are not attributed to either the origin or destination class. For example, in a parcel database, you may have a relationship class between parcels and owners in which owners own parcels and parcels are owned by owners. An attribute of that relationship may be percentage of ownership.



Subtypes toolset

Contains tools to manage the subtypes of a feature class or a table.

- ❖ **Add Subtype:** adds a new subtype to the set of subtypes in a feature class or table.

```
AddSubtype <in_table> <subtype_code> <subtype_description>
```

- If you add a subtype whose code already exists, the new subtype will be ignored.
- If you need to change the description of an existing subtype, you would first have to remove the subtype then add a new subtype with the same code and a new description.

- ❖ **Remove Subtype:** deletes a subtype from the set of associated subtypes in a feature class or table.

```
RemoveSubtype <in_table> <subtype_code;subtype_code...>
```

- Subtypes are removed using their integer code.
- The subtypes of a feature class or table can also be managed in ArcCatalog. Subtypes can be created and modified using the Subtypes Property page on the dataset's Properties dialog box.

❖ **Set Default Subtype:** sets the default subtype value.

`SetDefaultSubtype <in_table> <subtype_code>`

- The input table must contain subtype codes before setting a default code. Use Add Subtype and Set Subtype Field to create subtype codes.

❖ **Set Subtype Field:** defines the field in the feature class or table that stores the subtype codes.

`SetSubtypeField <in_table> <field>`

- A feature class or table can have only one subtype field.
- After a subtype field is set, subtype codes are added to the feature class or table with the Add Subtype function.



Table toolset

Contains tools to help you create and evaluate tabular data from a variety of sources.

◆ **Analyze:** updates relational database management system statistics for a dataset.

`Analyze <in_dataset> <BUSINESS | FEATURE | RASTER | ADDS | DELETES>`

- After data loading, deleting, updating, and compressing operations, it is important to update RDBMS statistics in Oracle®, Microsoft® SQL Server, DB2®, or Informix® databases.
- The Analyze function updates the statistics of business tables, feature tables, raster tables, added tables, and deleted tables, along with the statistics on those tables' indexes.

◆ **Change Privileges:** changes the user privileges associated with a dataset.

`ChangePrivileges <in_dataset> <user> {AS_IS | GRANT | REVOKE} {AS_IS | GRANT | REVOKE}`

- To edit ArcSDE datasets, both the View and Edit parameters must be granted. Edit privileges are dependent on the View privilege, since you can't edit what you can't see (view).
- Edit privileges may be revoked, but you can still view the dataset. However, if the View privilege is revoked, the Edit privileges will automatically be revoked as well.
- Edit privileges should only be granted on a versioned database. To edit the dataset, the database must be versioned and you must have both View and Edit privileges.
- The relational database management system (RDBMS) equivalent command for the View parameter is Select.
- The RDBMS equivalent commands for the Edit parameter are Update, Insert, and Delete. All three are granted or revoked simultaneously by the Edit parameter.

❖ **Copy Rows:** copies rows from a feature class, layer, table, or table view.

`CopyRows <in_rows> <out_table> {configuration_keyword}`

- If the input rows are a feature class, then only the attributes, and not the geometry, will be copied to the output table.

❖ **Create Table:** creates an empty geodatabase or dBASE table.

`CreateTable <out_path> <out_name> {template;template...} {configuration_keyword}`

❖ **Delete Rows:** deletes rows from a feature class, layer, table, or table view.

`DeleteRows <in_rows>`

- The input rows can be an INFO table, dBASE file, ArcSDE or personal geodatabase feature class or table, shapefile, layer, or table view.
- If Delete Rows is used on feature data, the entire row, including the geometry, will be deleted.

◆ **Get Count:** returns the number of rows in the feature class, layer, table, or table view.

`GetCount <in_rows>`

- The row count returned by the tool will be displayed in the geoprocessing window.
- If the input contains a selected set of records, only the selected records will be counted.
- This tool can be used in ModelBuilder to set up a precondition. Get Count is used to check the number of records returned by Select. If the record count is zero, then Buffer will not run.

Pivot Table: sorts and summarizes table fields, based on selected fields, to reduce redundancy.

`PivotTable <in_table> <fields;fields...> <pivot_field> <value_field> <out_table>`

Input Table

EntID	LinkType	TableID	LinkValue
1	A	X1	20
1	A	X2	21
2	A	X1	23
2	A	X2	29
2	B	X1	80
2	B	X2	77

Input Fields Pivot Field Value Field

Output Table

EntID	LinkType	X1	X2
1	A	20	21
2	A	23	29
2	B	80	77

- Pivot Table is used to reduce redundant records and flatten one-to-many relationships.
- If the pivot field is a numeric type, its value will be appended to its original field name in the output table.



Topology toolset

Contains tools to establish and manage topological relationships between features.

◆ **Add Feature Class To Topology:** adds a new feature class to a topological relationship.

`AddFeatureClassToTopology <in_topology> <in_feature_class> <xy_rank> <z_rank>`

- Adding a new rule to a topology automatically makes the entire topology dirty, so when you finish adding feature classes and rules, you will need to revalidate the topology. The new features may create errors, depending on the rules that you add.
- The input topology cannot be registered as versioned.
- The input feature class cannot be registered as versioned.

◆ **Add Rule To Topology:** adds a rule to the management of the topological relationship within a feature dataset.

`AddRuleToTopology <in_topology> <Must Not Have Gaps (Area) | Must Not Overlap (Area) | Must Be Covered By Feature Class Of (Area-Area) | Must Cover Each Other (Area-Area) | Must Be Covered By (Area-Area) | Must Not Overlap With (Area-Area) | Must Be Covered By Boundary Of (Line-Area) | Must Be Covered By Boundary Of (Point-Area) | Must Be Properly Inside (Point-Area) | Must Not Overlap (Line) | Must Not Intersect (Line) | Must Not Have Dangles (Line) | Must Not Have Pseudo-Nodes (Line) | Must Be Covered By Feature Class Of (Line-Line) | Must Not Overlap With (Line-Line) | Must Be Covered By (Point-Line) | Must Be Covered By Endpoint Of (Point-Line) | Boundary Must Be Covered By (Area-Line) | Boundary Must Be Covered By Boundary Of (Area-Area) | Must Not Self-Overlap (Line) | Must Not Self-Intersect (Line) | Must Not Intersect Or Touch Interior (Line) | Endpoint Must Be Covered By (Line-Point) | Contains Point (Area-Point) | Must Be Single Part (Line)> <in_feature_class> {subtype} {in_feature_class2} {subtype2}`

- To add a rule to a topology, the input topology cannot be registered as versioned.
- You can type the name of the subtype value to which you want a topology rule to be applied. If you are using the command line and the subtype name consists of more than one word, use single quotes to contain the subtype name.

◆ **Create Topology:** creates a topology within a feature dataset.

`CreateTopology <in_dataset> <out_name> <in_cluster_tolerance>`

- The topology has a minimum and maximum cluster tolerance that is derived from the precision of the spatial reference of the feature dataset in which you are creating the topology. If the value entered is larger than the maximum cluster tolerance, the value will become the maximum. If the value entered is smaller than the minimum, the minimum (or default) will be used.

◆ **Remove Feature Class From Topology:** removes a feature class from the topological relationship.

`RemoveFeatureClassFromTopology <in_topology> <in_feature_class>`

- Removing a feature class from a topology also removes all the topology rules associated with that feature class.
- Removing a feature class from a topology will invalidate the entire topology.
- Removing a feature class from a topology alters the schema of the topology. To remove a feature class, the topology must be unregistered as versioned.

◆ **Remove Rule From Topology:** removes a rule from a topological relationship.

`RemoveRuleFromTopology <in_topology> <in_rule>`

- When removing a rule from a topology using the command line or scripting, you must specify the feature class ObjectClassID in brackets after the rule name. See the command line syntax for an example. If you are using the Remove Feature Class From Topology tool, the list of rules is presented to you in a dropdown list.
- Removing a rule will require the topology to be revalidated.
- Removing a topology rule alters the schema of the topology and requires an exclusive lock. It also requires that the topology be unregistered as versioned.

◆ **Set Cluster Tolerance:** alters the cluster tolerance in a topological relationship.

`SetClusterTolerance <in_topology> <cluster_tolerance>`

- You cannot alter the cluster tolerance for a topology if the topology has been registered as versioned.
- Changing the cluster tolerance will require the entire topology be validated.

◆ **Validate Topology:** evaluates the features against the rules and finds any new errors related to new rules or feature classes.

`ValidateTopology <in_topology> {FULL_EXTENT | VISIBLE_EXTENT}`

- If you validate an ArcSDE geodatabase topology in ArcCatalog, the feature dataset that the topology is within must not be registered as versioned.
- Validate will only process areas of the topology that require validation. If you validate the topology in ArcMap, you can optionally use the visible extent as the area you want to validate. All areas outside the visible extent will not be validated.



Versions toolset

Contains tools to make adjustments to the versions of the data.

◆ **Alter Version:** alters the properties of any of the versions of the dataset including name, description, and access permissions.

`AlterVersion <in_workspace> <in_version> {name} {description} {PRIVATE | PUBLIC | PROTECTED}`

- Personal geodatabases don't support versioning. Versioning tools only work with ArcSDE data.

- Versions are not affected by changes occurring in other versions of the database.
- A version's permission can only be changed by its owner.

◆ **Create Version:** creates a new version of the specified database.

`CreateVersion <in_workspace> <parent_version> <version_name>`

- The output version name is prefixed by the SDE geodatabase user name; for example: SDE.arctoolbox.
- The output version's permissions are set as private by default but can be changed using the Alter Version tool.

◆ **Delete Version:** deletes the specified version from the input workspace.

`DeleteVersion <in_workspace> <version_name>`

- Only the version's owner can rename, delete, or alter the version.
- A parent version can't be deleted until all dependent child versions are deleted.
- Versions are not affected by changes occurring in other versions of the database.

◆ **Post Version:** applies the current edit session to the reconciled target version during versioned geodatabase editing.

`PostVersion <in_workspace> <version_name>`

- Posting synchronizes the edit version with the reconciled version and saves the data.
- Posting can't be undone since you are applying changes to a version that you are not currently editing.
- If the reconciled version is modified between reconciling and posting, you will be notified to reconcile again before posting.

◆ **Reconcile Version:** reconciles an ArcSDE version against a parent version in its lineage.

`ReconcileVersion <in_workspace> <version_name> <target_name>`

- The reconcile process requires that you are the only user currently editing the version and that you are the only user able to edit the version throughout the reconcile process until you save or post.
- The reconcile process requires that you have full permissions to all the feature classes that have been modified in the version being edited.

◆ **Register As Versioned:** registers an ArcSDE dataset as versioned in ArcCatalog.

`RegisterAsVersioned <in_dataset>`

- Registering a feature dataset as versioned registers all feature classes within the feature dataset as versioned.
- Versions are not affected by changes occurring in other versions of the database.
- A version's permission can only be changed by its owner.
- Making a feature class or table multiversioned requires a unique integer field. This is typically the OBJECTID field.

◆ **Unregister As Versioned:** unregisters a dataset as versioned in ArcCatalog.

`UnregisterAsVersioned <in_dataset> {KEEP_EDIT | NO_KEEP_EDIT}`

- Making a feature class or table multiversioned requires a unique integer field. Only the owner of the data can register or unregister the object as versioned.
- Versions are not affected by changes occurring in other versions of the database.
- A version's permission can only be changed by its owner.
- Outstanding edits will be lost with the execution of this tool unless the database is compressed.



Workspace toolset

Contains tools to create the storage models used with ArcGIS.

Create ArcInfo Workspace: creates a workspace with an INFO subdirectory.

`CreateArcInfoWorkspace <out_folder_path> <out_name>`

- The workspace cannot already exist.

❖ **Create Feature Dataset:** creates an empty feature dataset within an existing geodatabase.

`CreateFeatureDataset <out_dataset_path> <out_name> {spatial_reference}`

- A feature dataset is a collection of feature classes that share the same spatial reference.
- The spatial reference includes its coordinate system—either geographic or a specific projection—and the coordinate domains.

❖ **Create Folder:** creates a new folder.

`CreateFolder <out_folder_path> <out_name>`

- The output folder should not already exist. An error will not occur if the folder already exists.

❖ **Create Personal GDB:** creates a new personal geodatabase.

`CreatePersonalGDB <out_folder_path> <out_name>`

- The output personal geodatabase cannot already exist.
- Geodatabase names must start with a valid letter. The first character cannot be numeric.
- A personal geodatabase is usually on the same network as the client application (for example, ArcCatalog) and supports one editor at a time.

Geocoding toolbox

Contains tools used to manage a geocoding service and run geocoding actions.

- ◆ **Automate Geocoding Indexes:** creates an automatically updating relationship between the reference data and the geocoding index(es) of an address locator.

`AutomateGeocodingIndexes <in_address_locator>`

- This tool works only for address locators that use geodatabase reference data. Using this tool with locators that use other types of reference data, such as shapefiles, will generate an error.
- You must have write privileges on the reference feature class(es) and geocoding index file(s) or table(s) used by the address locator to use this tool.
- Using this tool with an address locator whose geocoding indexes are already automated will generate an error.
- It's a good idea to use the Rebuild Geocoding Indexes tool before using the Automate Geocoding Indexes tool to ensure that the address locator's geocoding indexes are current before automating them.

- ❖ **Create Address Locator:** creates a new address locator.

`CreateAddressLocator <in_address_locator_style> <reference_data{Role};
reference_data{Role}...> <in_field_map> <out_address_locator>`

- When you choose an ArcSDE address locator style on which to base the new address locator, you must store the new address locator in the same ArcSDE database.
- A reference dataset defines the role that it plays as reference data for the address locator. The address locator styles provided with ArcGIS use the following values to describe the roles of reference datasets:
 - Primary table—Defines the primary reference dataset feature class for a locator, such as a street centerline feature class
 - Alternate Name table—Defines an alternate street name table if the address locator style supports it
 - Alias table—Defines a place-name alias table

- ◆ **Deautomate Geocoding Indexes:** removes the automatically updating relationship between the reference data and geocoding index of an address locator.

`DeautomateGeocodingIndexes <in_address_locator>`

- This tool works only for address locators that use geodatabase reference data. Using this tool with locators that use other types of reference data, such as shapefiles, will generate an error.
- You must have write privileges on the reference feature class(es) and geocoding index files(s) or table(s) used by the address locator to use this tool.
- Using this tool with an address locator whose geocoding indexes are not automated will generate an error.

- ❖ **Delete Address Locator:** deletes an address locator.

`DeleteAddressLocator <in_address_locator>`

- You must have write privileges to the locator to delete it. For local locators, this means that you must have write access to the folder that contains the locator. For ArcSDE locators, this means that you must be the owner of the locator.
- Deleting an address locator does not delete the geocoding indexes used by the locator.

- ❖ **Geocode Addresses:** creates a point feature class from a table of addresses.

`GeocodeAddresses <in_table> <address_locator> <in_address_fields> <out_feature_class> {STATIC | DYNAMIC}`

- The input address table can be any format supported by ArcGIS, including INFO, dBASE, and geodatabase tables.
- The output feature class can be a shapefile or geodatabase feature class.
- Choose to create a dynamic geocoded feature class if you want edits in the input address table to be automatically reflected in the output feature class. This option is only valid if the input address table and output feature class are in the same geodatabase.

❖ **Rebuild Geocoding Indexes:** rebuilds the indexes of an address locator.

`RebuildGeocodingIndex <in_address_locator>`

- You must have write privileges on the geocoding index files or table(s) used by the address locator to use this tool.
- If the geocoding indexes used by the address locator don't exist, the Rebuild Geocoding Indexes tool will create them.

❖ **Standardize Addresses:** standardizes the address information in a table or feature class.

`StandardizeAddresses <in_address_data> <in_input_address_fields;in_input_address_fields...> <in_address_locator_style> <in_output_address_fields;in_output_address_fields...> <out_address_data> {STATIC | DYNAMIC}`

- If you are using an address locator style that geocodes addresses with house numbers (most address locator styles do this), you must include a field that contains a numeric value that represents the house number in the input address fields.
- If your input table or feature class and output table are in the same geodatabase, you can choose to create dynamic output.
- By default, the Standardize Addresses tool will create static output, unless you explicitly specify to create dynamic output.

Linear Referencing toolbox

Contains tools to model relative locations along linear features and associate multiple sets of attributes to portions of linear features.

❖ **Calibrate Routes:** adjusts route measures by reading measure information stored as an attribute in a point feature class.

```
CalibrateRoutes <in_route_features> <route_id_field> <in_point_features> <point_id_field>
<measure_field> <out_feature_class> {DISTANCE | MEASURES} {search_radius} {BETWEEN |
NO_BETWEEN} {BEFORE | NO_BEFORE} {AFTER | NO_AFTER} {IGNORE | NO_IGNORE} {KEEP | NO_KEEP}
{INDEX | NO_INDEX}
```

- If the output route feature class will be written to a geodatabase, an appropriate M Domain should be set.
- The output route feature class will include all the fields from the input features.
- A search radius of infinity cannot be specified.

❖ **Create Routes:** creates routes from existing lines.

```
CreateRoutes <in_line_features> <route_id_field> <out_feature_class> <LENGTH | ONE_FIELD
| TWO_FIELDS> <from_measure_field> <to_measure_field> {UPPER_LEFT | LOWER_LEFT |
UPPER_RIGHT | LOWER_RIGHT} {measure_factor} {measure_offset} {IGNORE | NO_IGNORE}
{INDEX | NO_INDEX}
```

- If the output route feature class will be written to a geodatabase, an appropriate M Domain should be set.
- The unique values in the route identifier field are written to output route feature class.
- Use Make Feature Layer or Make Query Table to effectively reduce the number of lines that will be used to create routes.
- Use a measure factor to convert between route measure units. For example, to convert from feet to miles, use a factor of 0.00018939394.

❖ **Dissolve Route Events:** removes redundant information from event tables or separates event tables having more than one descriptive attribute into separate tables.

```
DissolveRouteEvents <in_events> <in_event_properties> <dissolve_field; dissolve_field...>
<out_table> <out_event_properties> {DISSOLVE | CONCATENATE} {INDEX | NO_INDEX}
```

- The output table can be a dBASE file or a geodatabase table.
- If the input events do not have an ObjectID field, use Make Query Table prior to Dissolve Route Events to add a virtual ObjectID field.
- An attribute index on the route identifier field speeds up the dynamic segmentation process. If you will be using the output route feature class for dynamic segmentation, it is recommended that you choose to have an attribute index created.

❖ **Locate Features Along Routes:** computes the intersection of input features (point, line, or polygon) and route features and writes the route and measure information to a new event table.

```
LocateFeaturesAlongRoutes <in_features> <in_routes> <route_id_field> <radius_or_
tolerance> <out_table> <out_event_properties> {FIRST | ALL} {DISTANCE | NO_DISTANCE}
{ZERO | NO_ZERO} {FIELDS | NO_FIELDS}
```

- The output table can be a dBASE file or a geodatabase table.
- The event type must be point when the input features are points and must be line when the input features are lines or polygons.
- Use Make Feature Layer to effectively reduce the inputs that will be processed.

❖ **Make Route Event Layer:** creates a temporary route event layer.

```
MakeRouteEventLayer <in_routes> <route_id_field> <in_table> <in_event_properties>  
<out_layer> {offset_field} {NO_ERROR_FIELD | ERROR_FIELD} {NO_ANGLE_FIELD | ANGLE_FIELD}  
{NORMAL | TANGENT} {ANGLE | COMPLEMENT} {LEFT | RIGHT} {POINT | MULTIPOINT}
```

- Temporary layers are stored in memory and can be used as input to other geoprocessing functions in your current ArcCatalog or ArcMap session.
- Use Make Feature Layer on the routes and/or Make Table View on the events prior to Make Route Event Layer to reduce the number of routes and events that will be processed.

❖ **Overlay Route Events:** combines two input event tables to create a single output event table, using either a union or intersection operation.

```
OverlayRouteEvents <in_table> <in_event_properties> <overlay_table> <overlay_event_  
properties> <INTERSECT | UNION> <out_table> <out_event_properties> {ZERO | NO_ZERO}  
{FIELDS | NO_FIELDS} {INDEX | NO_INDEX}
```

- Line on line, line on point, point on line, and point on point event overlays can be performed.
- The input and overlay events should be based on the same route reference.

❖ **Transform Route Events:** transforms the measures of events from one route reference to another and writes them to a new event table.

```
TransformRouteEvents <in_table> <in_event_properties> <in_routes> <route_id_field>  
<target_routes> <target_route_id_field> <out_table> <out_event_properties> <cluster_  
tolerance> {FIELDS | NO_FIELDS}
```


- Transforming events allows you to use the events from one route reference with another route reference having different route identifiers and/or measures.
- Any whole or partial event that intersects a target route is written to the new event table.
- The output event type (point or line) must match the input event type.
- The best results will be achieved when the source routes and the target routes closely overlay. Do not use a large cluster tolerance to try and overcome discrepancies between the source and target routes because it can lead to unexpected results.
- Use Make Table View prior to Transform Route Events to effectively reduce the number of events that will be processed.

Spatial Statistics toolbox

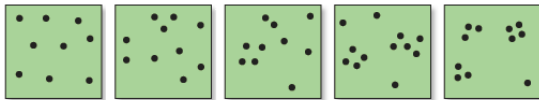
Contains statistical tools for analyzing the distribution of geographic features

Analyzing Patterns toolset

Contains tools to calculate statistical values used to quantify pattern.


-  **Average Nearest Neighbor:** calculates a nearest neighbor index based on the average distance from each feature to its nearest neighboring feature.

`AverageNearestNeighbor <input_feature_class> <EUCLIDEAN DISTANCE | MANHATTAN DISTANCE> {FALSE | TRUE} {are}`

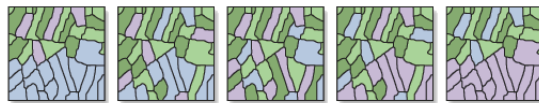


Dispersed ← → Clustered

- The nearest neighbor index is expressed as the ratio of the observed distance divided by the expected distance (hypothetical random distribution). Hence, if the index is less than 1, the pattern exhibits clustering; if the index is greater than 1, the trend is toward dispersion or competition.
- If an area value is not specified, then a value calculated from the map extent is used ($\text{area} = (\text{xmax} - \text{xmin}) * (\text{ymax} - \text{ymin})$). The nearest neighbor function, however, is sensitive to the area value (small changes in the area can result in significant changes in the results). To get better results, an accurate value for area should be used.


-  **High/Low Clustering (Getis-Ord General G):** measures concentrations of high or low values for a study area.

`HighLowClustering <input_feature_class> <input_field> {FALSE | TRUE} <INVERSE DISTANCE | INVERSE DISTANCE SQUARED | FIXED DISTANCE BAND | ZONE OF INDIFFERENCE | GET SPATIAL WEIGHTS FROM FILE> <EUCLIDEAN DISTANCE | MANHATTAN DISTANCE> <NONE | ROW | GLOBAL> <distance_band_or_threshold_distance> {weights_matrixfile}`

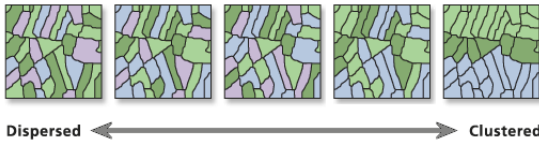


Low Cluster ← → High Cluster

- A high General G value indicates that high values are clustered within the study area; a low General G value indicates that low values tend to cluster.
- Although this tool will work with polygon or line data, it is really only appropriate for event, incident, or other fixed-point feature data.
- For line and polygon features, geometric centroids are calculated before the central feature is identified. The geometric centroid of a feature may be located outside a feature's boundary. If centroids must be within feature boundaries, use the Features To Point (Inside option) tool to create centroids before performing the nearest neighbor operation.

-  **Spatial Autocorrelation (Moran's I):** measures spatial autocorrelation based on feature locations and attribute values.

`SpatialAutocorrelation <input_feature_class> <input_field> {FALSE | TRUE} <INVERSE DISTANCE | INVERSE DISTANCE SQUARED | FIXED DISTANCE BAND | ZONE OF INDIFFERENCE | GET SPATIAL WEIGHTS FROM FILE> <EUCLIDEAN DISTANCE | MANHATTAN DISTANCE> <NONE | ROW | GLOBAL> <distance_band_or_threshold_distance> {weights_matrixfile}`



- Given a set of features and an associated attribute, Global Moran's I evaluates whether the pattern expressed is clustered, dispersed, or random. A Moran's I value near +1.0 indicates clustering; a value near -1.0 indicates dispersion.



Mapping Clusters toolset

Contains tools for cluster analysis, such as identifying the locations of statistically significant hot spots or areas of significant diversity.



- ❖ **Cluster And Outlier Analysis (Anselin Local Moran's I):** identifies clusters of points with values similar in magnitude and clusters of points with very heterogeneous values within a set of weighted data points.

`ClustersOutliers <input_feature_class> <input_field> <output_feature_class> <INVERSE DISTANCE | INVERSE DISTANCE SQUARED | FIXED DISTANCE BAND | ZONE OF INDIFFERENCE | GET SPATIAL WEIGHTS FROM FILE> <EUCLIDEAN DISTANCE | MANHATTAN DISTANCE> <NONE | ROW | GLOBAL> <distance_band_or_threshold_distance> {weightsmatrix_file}`

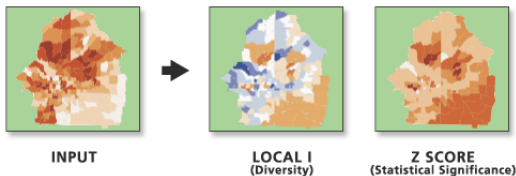


- If the I index value is positive, then that feature has values similar to neighboring features' values. If the I index value is negative, then that feature is quite different from neighboring values.
- The Local Moran's I process copies the input feature class to the output feature class and adds two results columns for the index and z score named `LMi<distance_method>` and `LMz<distance_method>`. If fields of these names already exist in the input feature class, they will be overwritten in the output feature class.



- ❖ **Cluster/Outlier Analysis With Rendering:** identifies clusters of points with values similar in magnitude and clusters of points with very heterogeneous values within a given set of weighted data points and applies a cold to hot type of rendering.

`ClustersOutliersRendered <input_feature_class> <input_field> <output_layer_file> <output_feature_class>`

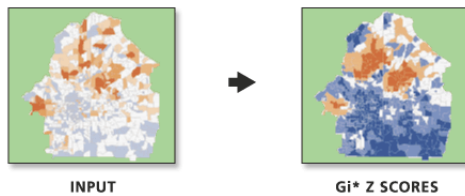


- The Cluster/Outlier Analysis With Rendering tool combines the Cluster and Outlier Analysis and Z Score Rendering functions.




- ❖ **Hot Spot Analysis (Getis-Ord Gi*):** identifies spatial clusters of statistically significant high or low attribute values by calculating the Getis-Ord Gi* statistics.

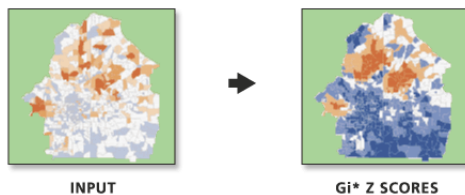
`HotSpot <input_feature_class> <input_field> <output_feature_class> <FIXED DISTANCE BAND | INVERSE DISTANCE | INVERSE DISTANCE SQUARED | ZONE OF INDIFFERENCE | GET SPATIAL WEIGHTS FROM FILE> <EUCLIDEAN DISTANCE | MANHATTAN DISTANCE> <NONE | ROW | GLOBAL> <distance_band_or_threshold_distance> {self_potential_field} {weights_matrix_file}`



- Given a set of weighted data points, the Getis-Ord Gi* statistic identifies those clusters of points with values higher in magnitude than you might expect to find by random chance. (For line and polygon features, centroids are calculated prior to analysis.)
- The Gi function creates a new feature class that duplicates the input feature class, then adds a new Results column for the Gi z score. The name of the output field is Gi<distance_method>. If a field of this name already exists in the input feature class, it will be overwritten in the output feature class.

 **Hot Spot Analysis with Rendering:** calculates Gi* statistics and applies a cold to hot type of rendering to the output Z scores.


`HotSpotsRendered <input_feature_class> <input_field> <output_layer_file> <output_feature_class> <distance_and_or_threshold_distance>`



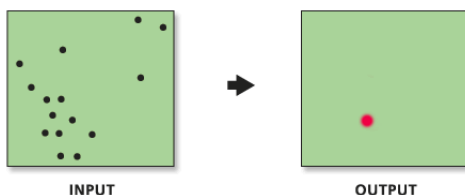
- The Gi* rendered model combines the functions Hot Spot Analysis and Z Score Rendering.

Measuring Geographic Distributions toolset

Contains tools to calculate a value that represents a characteristic of the distribution of a set of features, such as the center, compactness, or orientation.

 **Central Feature:** identifies the most centrally located feature in a point, line, or polygon feature.

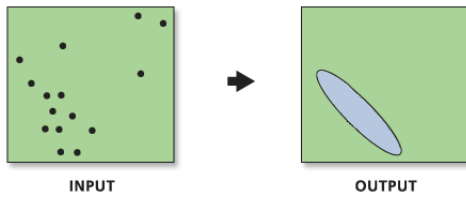
`CentralFeature <input_feature_class> <output_feature_class> <EUCLIDEAN DISTANCE | MANHATTAN DISTANCE> {weight_field}{self_potential_weight_field}`




- An origin-destination (O/D) distance matrix is created from feature centroids (using Features To Points to create the centroids and Point Distance to create the O/D matrix).
- The feature associated with the smallest O/D row sum is the most centrally located feature; this feature is selected and copied to the newly created output feature class (using Make Feature Layer and Save To Layer File).

 **Directional Distribution (Standard Deviational Ellipse):** measures whether a distribution of features exhibits a directional trend.

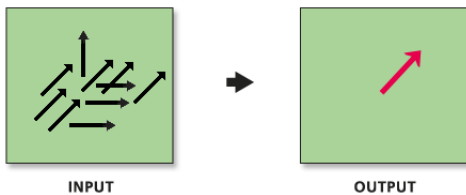
`DirectionalDistribution <input_feature_class> <output_ellipse_feature_class> <1 STANDARD DEVIATION | 2 STANDARD DEVIATIONS | 3 STANDARD DEVIATIONS> {weight_field} {case_field}`




- The following fields are added to the output ellipse feature class to store ellipse information: CenterX stores the x-coordinate of the center of the ellipse; CenterY stores the y coordinate of the center of the ellipse; XStdDist stores the standard distance in the x direction; and YStdDist stores the standard distance in the y direction and rotation, which stores the rotation of the ellipse (if you specify a case field, it is also included in the output feature class).

 **Linear Directional Mean:** identifies the general (mean) direction for a set of vectors.

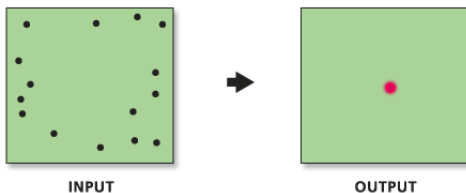
`DirectionalMean <input_feature_class> <output_feature_class> <orientation_only> {case_field}`




- Attribute values for the new line features include Compass Angle (clockwise from due north), Directional Mean (counterclockwise from due east), Circular Variance (an indication of how much directions or orientations deviate from directional mean), Mean Center X and Y Coordinates, and Mean Length.
- Analogous to a standard deviation measure, the circular variance tells how well the directional mean vector represents the set of input vectors. Circular variance ranges from 0 to 1. If all the input vectors have the exact same (or very similar) directions, the circular variance is small (near 0). When input vector directions span the entire compass, the circular variance is large (near 1).

 **Mean Center:** identifies the geographic center (or the center of concentration) for a set of features.

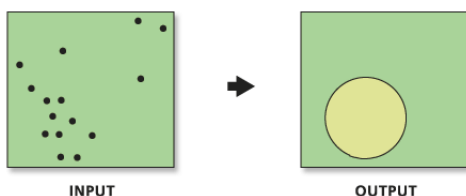
`MeanCenter <input_feature_class> <output_feature_class> {weight_field} {case_field} {dimension_field}`



- If a case field is specified, the input features are grouped according to case field values and a mean center is calculated from the average x- and y-values for the centroids in each group.

 **Standard Distance:** measures the degree to which features are concentrated or dispersed around the points (or feature centroids) in an input feature class.

`StandardDistance <input_feature_class> <output_standard_distance_feature_class> <1 STANDARD DEVIATION | 2 STANDARD DEVIATIONS | 3 STANDARD DEVIATIONS> {weight_field} {case_field}`





Utilities toolset

Contains tools to perform a variety of data rendering tasks that can be used in conjunction with other tools in the Spatial Statistics toolbox.

❖ **Calculate Areas:** Calculates area values for each feature in a polygon feature class.

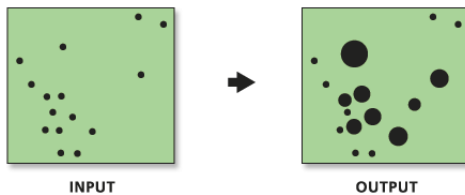
`CalculateAreas <input_feature_class> <output_feature_class>`

- The field F_AREA is created in the output feature class to store calculated area values. If a field of this name already exists in the input feature class, it will be overwritten in the output feature class.



❖ **Collect Events:** collects event data into weighted point data.

`CollectEvents <input_incident_features> <output_weighted_point_feature_class>`

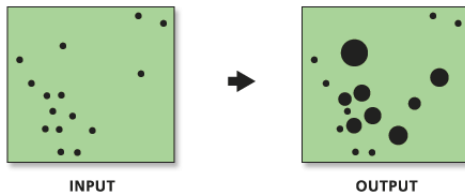


- Collect Events creates a new feature class and adds a field named Count to hold the sum of all incidents for each unique location. If a field of this name already exists in the input feature class, it will be overwritten in the output feature class.



❖ **Collect Events With Rendering:** collects event data into weighted point data and applies a graduated circle rendering to the Count field.

`CollectEventsRendered <input_incident_features> <output_layerfile> <output_weighted_point_feature_class>`

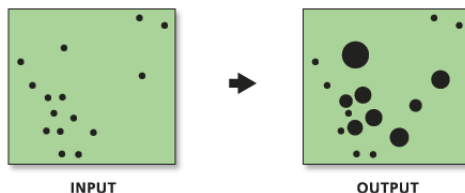


- The Collect Events Rendered model combines the functions Collect Events and Count Rendering.



❖ **Count Rendering:** applies graduated circle rendering to a count type field of a point feature class.


`CountRenderer <input_feature_class> <field_to_render> <output_layer_file> <number_of_classes> <MANGO | BRIGHT_RED | DARK_GREEN | GREEN | DARK_BLUE | BRIGHT_PINK | LIGHT_YELLOW | SKY_BLUE> {maximum_field_value}`



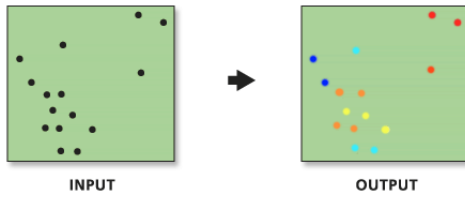
- The Count Renderer draws quantities using circle size to show relative values.

❖ **Export Feature Attribute To ASCII:** Exports feature class coordinates and attribute values to a space-, comma-, or semicolon-delimited ASCII text file.

`ExportXYv <input_feature_class> <value_field> <SPACE | COMMA | SEMI-COLON> <output_ascii_file>`

 **Z Score Rendering:** applies a cold to hot graduated color rendering to a field of z scores.

`ZRenderer <input_feature_class> <field_to_render> <output_layer_file>`



- Z Score Rendering is appropriate for rendering output from both Hot Spot Analysis and Diversity Analysis.

ArcGIS Desktop extensions

Geoprocessing tools



3D Analyst toolbox

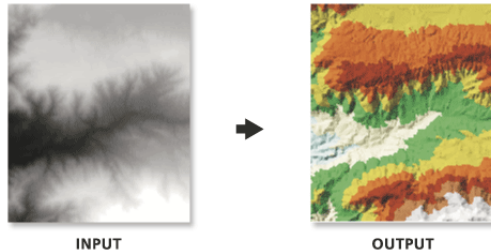
Contains tools to create and modify TIN and raster surfaces, then extract information and features from them.

Conversion toolset

Contains tools used to convert to and from a TIN.

Raster To TIN: creates a TIN from a raster dataset.

`RasterTin <in_raster> <out_tin> {z_tolerance} {max_points} {z_factor}`



- The default maximum allowable difference between the height of the input raster and the height of the output TIN is 1/10 of the z range of the input raster. This can be too large in many cases and should be adjusted to a smaller value. Values that are too small, on the other hand, can produce TINs that are too large for practical use. Specify a value as large as possible for the requirements of your project.

TIN Domain: extracts the interpolation zone from an input TIN into an output feature class.

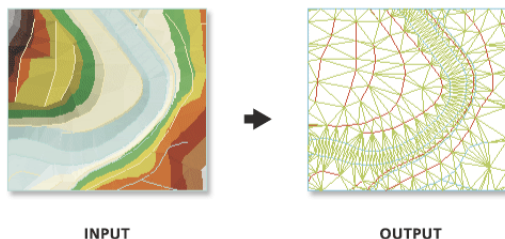
`TinDomain <in_tin> <out_feature_class> <LINE | POLYGON>`



- To create a polygon feature class based on the TIN's area, use the POLYGON option.
- To create a polyline feature class based on the TIN's extent, use the LINE option.

TIN Edge: extracts the triangle edge from an input TIN into an output feature class.

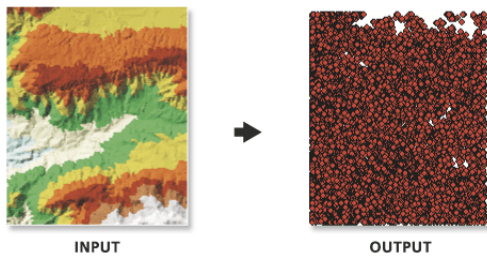
`TinEdge <in_tin> <out_feature_class> {DATA | SOFT | HARD | ENFORCED | REGULAR | OUTSIDE | ALL}`



- Use the optional {edge_type} argument to extract a specific type of triangle edge.

TIN Node: extracts nodes from an input TIN into an output feature class.

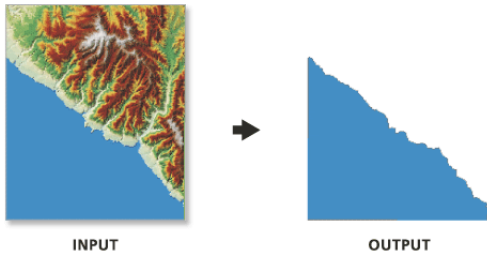
`TinNode <in_tin> <out_feature_class> {spot_field} {tag_field}`



- Indicate a `spot_field` to create a 2D feature class.
- Omit `spot_field` to create a 3D feature class.

TIN Polygon Tag: extracts polygon tag information from an input TIN into an output feature class.

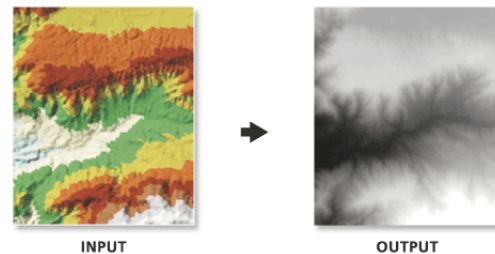
`TinPolygonTag <in_tin> <out_feature_class> {tag_field}`



- Polygons will be generated for all TIN triangles that have tag values.

TIN To Raster: converts a TIN to a raster.

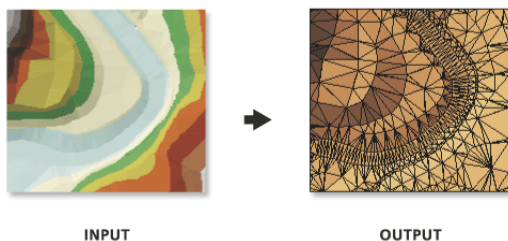
`TinRaster <in_tin> <out_raster> {FLOAT | INT} {LINEAR | NATURAL_NEIGHBORS} {OBSERVATIONS 250 | CELLSIZE 10} {z_factor}`



- The `data_type` option determines the data type of the output raster. Floating point preserves the accuracy of z-values, possibly at the expense of disk storage space. On average, integer grids require less storage space than floating-point grids when dealing with terrain data. If the accuracy requirements of your z-values are such that they can be represented by integer data, you may want to consider using the INT option.

TIN Triangle: extracts triangles as polygons from an input TIN into an output feature class.

`TinTriangle <in_tin> <out_feature_class> {PERCENT | DEGREE} {z_factor} {HILLSHADE} {tag_field}`



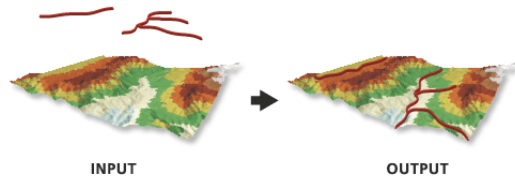


Functional Surface toolset

Contains tools to produce output providing knowledge about height information that is contained in surfaces.

Interpolate Shape: interpolates z-values for a feature class based on an underlying surface.

`InterpolateShape <in_surface> <in_feature_class> <out_feature_class> {sample_distance} {z_factor}`



- Choose a smaller sampling distance to increase the accuracy at which the feature class is interpolated.

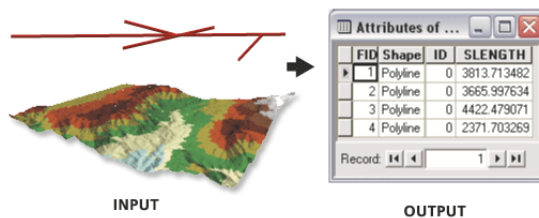
Line Of Sight: calculates the visibility across a surface between points.

`LineOfSight <in_surface> <in_line_feature_class> <out_los_feature_class> {out_obstruction_feature_class} {use_curvature} {use_refraction} {refraction_factor}`

- The observer is the point from which visibility is determined. The target is the opposite end of the line to which visibility is determined.
- Only the endpoints of the input line are used to determine target visibility.

Surface Length: calculates the surface length of each line in a feature class based on a functional surface.

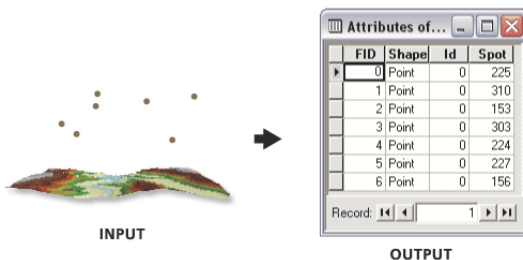
`SurfaceLength <in_surface> <in_feature_class> {out_length_field} {sample_distance} {z_factor}`



- Use a smaller sampling distance to increase the accuracy of the surface length calculations.
- Use {out_length_field} to give the length field a custom name.

Surface Spot: calculates surface values for each point of a point feature class by interpolating from a functional surface.

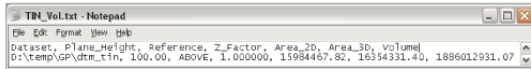
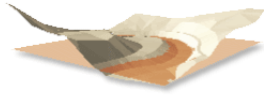
`SurfaceSpot <in_surface> <in_feature_class> {out_spot_field} {z_factor}`



- Ensure that the input surface and the input feature class have overlapping extents.
- Use the {z_factor} argument to convert the output field to the desired units.

Surface Volume: calculates the area and volume of a functional surface above or below a given reference plane.

`SurfaceVolume <in_surface> {out_text_file} {ABOVE | BELOW} {base_z} {z_factor}`



- To determine the volume of an isolated portion of a TIN surface, use the Add Feature Class To TIN tool to add a clip polygon defining your study area.
- Using a z-factor is essential for correct volume calculations when the surface z-units are expressed in a different unit of measure from the ground units. Using a z-factor does not modify the original data.



Raster Interpolation toolset

Contains tools to create a raster surface from point features.

IDW: interpolates a surface from points using an inverse distance weighted (IDW) technique.

`IDW <in_point_features> <z_field> <out_raster> {cell_size} {power} {search_radius}`
`{in_barrier_polyline_features}`

- The barriers option is used to specify the location of linear features known to interrupt the surface continuity. These features do not have z-values. Cliffs, faults, or embankments are typical examples of barriers. Barriers limit the selected set of the input sample points used to interpolate output z-values to those samples on the same side of the barrier as the current processing cell.
- The output value for a cell using IDW is limited to the range of the values used to interpolate. Because IDW is a weighted distance average, the average cannot be greater than the highest or less than the lowest input. Therefore, it cannot create ridges or valleys if these extremes have not already been sampled (Watson and Philip, 1985).
- The best results from IDW are obtained when sampling is sufficiently dense with regard to the local variation you are attempting to simulate. If the sampling of input points is sparse or uneven, the results may not sufficiently represent the desired surface (Watson and Philip, 1985).

Krige: interpolates a raster dataset from a set of points using kriging.

`Kriging <in_point_features> <z_field> <out_surface_raster> <semivariogram_props>`
`{cell_size} {search_radius} {out_variance_prediction_raster}`

- The universal kriging types (linear with linear drift and linear with quadratic drift) assume that there is a structural component present and that the local trend varies from one location to another.
- The advanced parameters allow control of the semivariogram used for kriging. A default value for Lag size is initially set to the default output cell size. For major range, partial sill, and nugget, a default value will be calculated internally if nothing is specified.
- Low values within the output variance of prediction raster indicate a high degree of confidence in the predicted value. High values may indicate a need for more data points.

Natural Neighbor: interpolates a surface from points using a natural neighbor technique.

`NaturalNeighbor <in_point_features> <z_field> <out_raster> {cell_size}`

- The Natural Neighbor tool can efficiently handle large numbers of input points. Other interpolators may have difficulty with large point datasets.

Spline: interpolates a surface from points using a minimum curvature spline technique.

```
spline <in_point_features> <z_field> <out_raster> {cell_size}
      {REGULARIZED | TENSION} {weight} {number_points}
```

- The resulting smooth surface from Spline passes exactly through the input points.
- The REGULARIZED option of Spline usually produces smoother surfaces than those created with the TENSION option.
- For the REGULARIZED option, higher values used for the Weight parameter produce smoother surfaces. The values entered for this parameter must be equal to or greater than zero. Typical values that are used are 0, 0.001, 0.01, 0.1, and 0.5. The weight is the square of the parameter, referred to in the literature as tau (τ).
- For the TENSION option, higher values entered for the Weight parameter result in somewhat coarser surfaces but with surfaces that closely conform to the control points. The values entered must be equal to or greater than zero. Typical values are 0, 1, 5, and 10. The Weight is the square of the parameter, referred to in the literature as phi (Φ).
- The greater the value of Number of Points, the smoother the surface of the output raster.

Topo To Raster: generates a hydrologically correct raster dataset of elevation.

```
TopoToRaster <feature_layer{Field} {Type};feature_layer{Field} {Type}...> <out_surface_
raster> {cell_size} {extent} {Margin} {minimum_z_value} {maximum_z_value} {ENFORCE |
NO_ENFORCE | ENFORCE_WITH_SINK} {CONTOUR | SPOT} {maximum_iterations} {roughness_
penalty} {discrete_error_factor} {vertical_standard_error} {tolerance_1} {tolerance_2}
{out_stream_features} {out_sink_features} {out_diagnostic_file} {out_parameter_file}
```

- Topo to Raster will only use four input data points for the interpolation of each output cell. All additional points are ignored. If too many points are encountered by the algorithm, an error may occur indicating the point dataset has too many points. The maximum number of points that can be used is $nrows * ncols$, where $nrows$ is the number of rows in the output raster and $ncols$ is the number of columns.
- Stream data always takes priority over point or contour data; therefore, elevation data points that conflict with descent down each stream are ignored. Stream data is a powerful way of adding topographic information to the interpolation, further ensuring the quality of the output DEM.
- Some typical values for the Tolerance 1 and Tolerance 2 settings are:
 - For point data at 1:100,000 scale, use 5.0 and 200.0.
 - For less dense point data at up to 1:500,000 scale, use 10.0 and 400.0.
 - For contour data with contour spacing of 10, use 5.0 and 100.0.

Topo To Raster By File: generates a hydrologically correct raster dataset of elevation.

```
TopoToRasterByFile <in_parameter_file> <out_surface_raster> {out_stream_features}
      {out_sink_features}
```

- The parameter file is structured with the input datasets listed first, followed by the various parameter settings, then the output options.

The input data identifies the input datasets and, where applicable, fields. There are six types of input: Contours, Points, Sinks, Streams, Lakes, and Boundaries. As many inputs can be used as desired, within reason. The order in which the inputs are entered does not have any bearing on the outcome. <Path> indicates a path to a dataset, <Item> indicates a field name, and <#> indicates a value to be entered.

- Contours—Contour line dataset with item containing height values.
- Points—Point dataset with item containing height values.
- Sinks—Point dataset containing sink locations. If the dataset has elevation values for the sinks, specify that field name as the <Item>. If only the locations of the sinks are to be used, use NONE for <Item>.
- Streams—Stream line dataset. Height values are not necessary.

- Lakes—Lake polygon dataset. Height values are not necessary.
- Boundary—Boundary polygon dataset. Height values are not necessary.
- Enforce—Controls whether drainage enforcement is applied.
- Datatype—Primary type of input data.
- Iterations—The maximum number of iterations the algorithm performs.
- Roughness Penalty—The measure of surface roughness.
- Discretization Error Factor—The amount to adjust the data smoothing of the input data into a raster.
- Vertical Standard Error—The amount of random error in the z-values of the input data.
- Tolerances—The first reflects the accuracy of elevation data in relation to surface drainage, and the other prevents drainage clearance through unrealistically high barriers.
- Z-Limits—Lower and upper height limits.
- Extent—Minimum x-, minimum y-, maximum x-, and maximum y-coordinate limits.
- Cell Size—The resolution of the final output raster.
- Margin—Distance in cells to interpolate beyond the specified output extent and boundary.
- Output Stream Features—Use only if Output stream polyline features is set in the Topo To Raster By File dialog box.
- Output Sink Features—Use only if Output remaining sink point features is set in the Topo To Raster By File dialog box.
- Output Diagnostics File—The location and name of the diagnostics file.
- An example parameter file is:


```
Contour D:\data\contours2\arc HEIGHT
Point D:\data\points2\point SPOTS
Sink D:\data\sinks_200.shp
Stream D:\data\streams\arc
Lake D:\data\lakes\polygon
Boundary D:\data\clipcov\polygon
ENFORCE ON
DATATYPE CONTOUR
ITERATIONS 40
ROUGHNESS_PENALTY 0.0000000000
DISCRETE_ERROR_FACTOR 1.0000000000
VERTICAL_STANDARD_ERROR 0.0000000000
TOLERANCES 2.5000000000 100.0000000000
ZLIMITS -2000.0000000000 13000.0000000000
EXTENT -810480.62500000000 8321785.0000000000 810480.62500000000
10140379.0000000000
CELL_SIZE 1800.0000000000
MARGIN 20
OUT_STREAM
OUT_SINK
OUT_DIAGNOSTICS D:\data\ttr_diag.txt
```

Trend: interpolates a surface from points using a trend technique.

Trend <in_point_features> <z_field> <out_raster> {cell_size} {order} {LINEAR | LOGISTIC}

- As the order of the polynomial is increased, the surface being fitted becomes progressively more complex. A higher-order polynomial will not always generate the most accurate surface; it is dependent on the data.
- The optional root mean square (RMS) file output contains information on the RMS error of the interpolation. This information can be used to determine the best value to use for the polynomial order, by changing the order value until you get the lowest RMS error.
- For the type of regression LOGISTIC option of, the z-value field of input point features should have codes of 0 and 1.

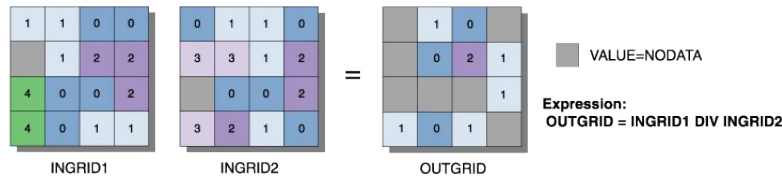


Raster Math toolset

Contains the tools used to perform mathematics with raster datasets.

Divide: divides the values of two inputs on a cell-by-cell basis.

Divide <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- The order of input is relevant for Divide.
- When a number is divided by 0, the output result is NoData.
- If both inputs are integers, then Divide performs an integer division and the output result is an integer. For example, if 3 is to be divided by 2, the output is 1.
- If either input is of floating-point type, then Divide performs a floating-point division and the result is a floating-point value. For example, if 3 is divided by 2.0, the output is 1.5.

Float: converts each cell value of a raster dataset to a floating-point value.

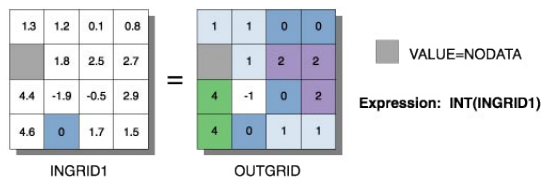
Float <in_raster_or_constant> <out_raster>



- Input values are integers and can be positive or negative.

Int: converts each cell value of a raster dataset into an integer value through truncation.

Int <in_raster_or_constant> <out_raster>



- If rounding is preferred rather than truncating, add a 0.5 input raster prior to performing Int.
- The difference between the Round Down and Int functions is that Int always truncates a number:
 - int on 1.5 becomes 1
 - int on -1.5 becomes -1
 while for the same two values, Round Down returns
 - round down on 1.5 becomes 1.0
 - round down on -1.5 becomes -2.0

A second difference between the two functions is that Round Down outputs floating-point values, and Int outputs integer values.

Minus: subtracts the values of the second input from the values of the first input.

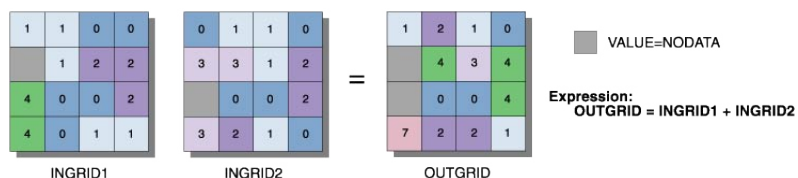
Minus <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- The order of input is relevant in the subtraction expression.

Plus: adds the values of two raster datasets on a cell-by-cell basis.

Plus <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- The order of input is irrelevant in the addition expression.

Times: multiplies the values in raster datasets on a cell-by-cell basis.

Times <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- The order of input is irrelevant in the multiplication expression.



Raster Reclass toolset

Contains tools to alter the classes within a raster dataset.

Lookup: creates a new raster dataset by looking up values found in another field in the table of the input raster dataset.

Lookup <in_raster> <lookup_field> <out_raster>

- If the lookup field is a numeric type, the values of that field will be written to the output raster attribute table as Value. Other items in the input raster attribute table will not be transferred to the output raster attribute table.

For example, an attribute table of input raster with numeric field Attr1:

Value	Count	Attr1
1	294	1
2	345	8
3	654	3

Output attribute table from Lookup on Attr1 field:

Value	Count
1	294
3	654
8	345

Reclass By ASCII File: reclassifies (or changes) the values of the input cells of a raster dataset by using an ASCII remap file.

`ReclassByASCIIFile <in_raster> <in_remap_file> <out_raster> {DATA | NODATA}`

- The output raster will always be of integer type. If the output assignment values in the ASCII file are floating-point values, an error message will be returned and the program will halt.

Reclass By Table: reclassifies (or changes) the values of the input cells of a raster dataset by using a remap table.

`ReclassByTable <in_raster> <in_remap_table> <from_value_field> <to_value_field>
<output_value_field> <out_raster> {DATA | NODATA}`

- The from value field, to value field, and output value field are the field names in the table that define the remapping.
- To reclassify individual values, use a simple remap table of two items. The first item identifies the value to reclassify, and the other item the value to assign it. Set the to value field to the same as the from value field. The value to assign to the output is Output value field.
- To reclassify ranges of values, the remap table must have items defining the start and end of each range, along with the value to assign the range. The item defining the start of the range is the from value field, and the value defining the end of the range is the to value field. The value to assign to the output is output value field.
- The remap table can be an INFO table, a .dbf file, an Access table, or a text file.
- The values in the from and to fields can be any numerical item. The assignment values in the output field must be integers.
- Values in the from field of the table must be sorted in ascending order and should not overlap.

Reclassify: reclassifies (or changes) the value in a raster dataset.

`Reclassify <in_raster> <reclass_field> <remap> <out_raster> {DATA | NODATA}`

- The remap table can be stored with the Save button. The Load button allows previously created remap tables to be used. Only remap tables created by the tool should be used in Reclassify.
- If running the Reclassify tool as part of a model within a ModelBuilder window, run the tools before the Reclassify tool in the model first. This will allow the values for the input raster to display properly in the Reclassification dialog box.

Slice: slices a range of values of the input cells by zones of equal area or equal interval.

`Slice <in_raster> <out_raster> <number_zones> {EQUAL_INTERVAL | EQUAL_AREA | NATURAL_BREAKS} {base_output_zone}`

- If a mask has been set, those cells that have been masked out will receive NoData on the output slice raster.

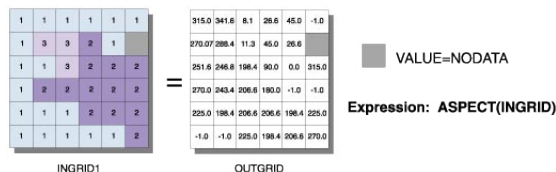


Raster Surface toolkit

Contains tools to analyze the surface of a raster dataset.

Aspect: identifies the direction of the maximum rate of change in z-value from each cell.

`Aspect <in_raster> <out_raster>`



- Aspect is the direction of the maximum rate of change in the z-value from each cell in a raster surface.
- Aspect is expressed in positive degrees from 0 to 359.9, measured clockwise from the north.
- Cells in the input raster of zero slope (for example, flat) are assigned an aspect of -1.

Contour: creates contours or isolines from a raster dataset surface.

`Contour <in_raster> <out_polyline_features> <contour_interval> {base_contour} {z_factor}`

- Smoother but less accurate contours may be obtained by first performing a neighborhood focal statistics operation with the mean option on the input raster.
- A base contour is used, for example, when you want to create contours every 15 meters, starting at 10 meters. Here, 10 would be used for the base contour and 15 would be the contour interval. The values to be contoured would be 10, 25, 40, 55, and so on.
- Specifying a base contour does not prevent contours from being created above or below that value.

Contour List: creates contours or isolines from a list of contour values.

`ContourList <in_raster> <out_polyline_features> <contour_values;contour_values...>`

- Smoother but less accurate contours can be obtained by first performing a neighborhood focal statistics operation with the mean option on the input raster.

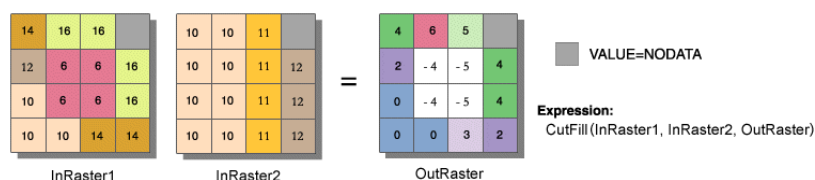
Curvature: calculates the curvature of a surface at each cell center.

`Curvature <in_raster> <out_curvature_raster> {z_factor} {out_profile_curve_raster} {out_plan_curve_raster}`

- The primary output is the curvature of the surface on a cell-by-cell basis, as fitted through that cell and its eight surrounding neighbors. Curvature is the second derivative of the surface, or the slope of the slope. Two optional output curvature types are possible; the profile curvature is in the direction of the maximum slope, and the plan curvature is perpendicular to the direction of the maximum slope.
- A positive curvature indicates that the surface is upwardly convex at that cell. A negative curvature indicates that the surface is upwardly concave at that cell. A value of zero indicates that the surface is flat.
- In the profile output, a negative value indicates that the surface is upwardly convex at that cell. A positive profile indicates that the surface is upwardly concave at that cell. A value of zero indicates that the surface is flat.
- In the plan output, a positive value indicates that the surface is upwardly convex at that cell. A negative plan indicates that the surface is upwardly concave at that cell. A value of zero indicates that the surface is flat.
- Units of the Curvature output raster, as well as the units for the optional output profile curve raster and output plan curve raster, are one over 100 z units, or 1/100 (z units). The reasonably expected values of all three output rasters for a hilly area (moderate relief) may differ from approximately -0.5 to 0.5; while for the steep, rugged mountains (extreme relief), the values may vary between -4 and 4.

Cut/Fill: calculates cut and fill areas.

`CutFill <in_before_surface> <in_after_surface> <out_raster> {z_factor}`

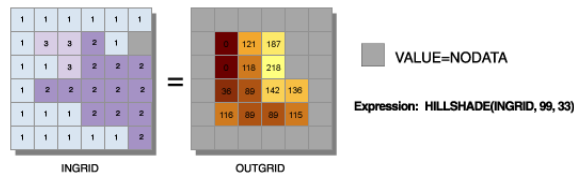


- The Cut/Fill tool enables you to create a map based on two input surfaces (before and after), displaying the areas and volumes of surface materials that have been modified by the addition or removal of surface material. Negative z-values indicate regions of the input before raster surfaces have been filled; positive regions indicate cuts.

- For accurate Cut/Fill results, the z units should be the same as the x,y ground units. This ensures that the resulting volumes are meaningful cubic measures (for example, cubic meters). If they are not the same, use a z-factor to convert z units to x,y units. For example, if your x,y units are meters and your z units are feet, you could specify a z-factor of 0.3048 to convert feet to meters.

Hillshade: creates a shaded relief raster dataset by considering the illumination angle and shadows.

Hillshade <in_raster> <out_raster> {azimuth} {altitude} {NO_SHADOWS | SHADOWS} {z_factor}



- The Hillshade tool creates a shaded relief raster from a raster. The illumination source is considered at infinity.
- Two types of shaded relief rasters can be output. Having model shadows unchecked outputs a raster that only considers the local illumination angle. Having model shadows checked outputs one that considers the effects of both the local illumination angle and shadow.

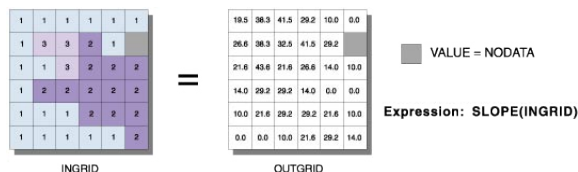
Observer Points: identifies exactly which observer points are visible from each surface location within the raster dataset.

ObserverPoints <in_raster> <in_observer_point_features> <out_raster> {z_factor} {FLAT_EARTH | CURVED_EARTH} {refractivity_coefficient}

- The visibility of each cell center is determined by comparing the altitude angle to the cell center with the altitude angle to the local horizon. The local horizon is computed by considering the intervening terrain between the point of observation and the current cell center. If the point lies above the local horizon, it is considered visible.

Slope: identifies the rate of maximum change in z-value from each cell.

Slope <in_raster> <out_raster> {DEGREE | PERCENT_RISE} {z_factor}



- Slope is the rate of maximum change in z-value from each cell.
- The use of a z-factor is essential for correct slope calculations when the surface z units are expressed in units different from the ground x,y units.
- Degree of slope is a value between 0 and 90.

Viewshed: determines the raster dataset surface locations visible to a set of observer features.

Viewshed <in_raster> <in_observer_features> <out_raster> {z_factor} {FLAT_EARTH | CURVED_EARTH} {refractivity_coefficient}

- The visibility of each cell center is determined by comparing the altitude angle to the cell center with the altitude angle to the local horizon. The local horizon is computed by considering the intervening terrain between the point of observation and the current cell center. If the point lies above the local horizon, it is considered visible.



TIN Creation toolset

Contains tools to create and edit TINs.

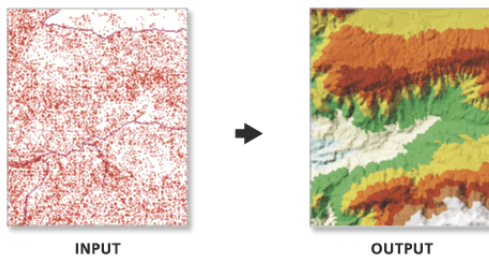
Create TIN: creates an empty TIN with an extent based on an input geodataset or coordinates.

`CreateTin <out_tin> <spatial_reference>`

- The output TIN cannot already exist.
- Use the Import button on the spatial reference dialog box to browse for, or when using the command line, indicate a dataset with a coordinate system close to that of the region of the TIN you are creating.
- To add features to the output TIN, use the Edit TIN tool.

Edit TIN: edits the TIN geoprocessing function.

`EditTin <in_tin> <in_feature_class> {Height_Field} {Tag_Field} {SF_Type} {Use_Z};
in_feature_class {Height_Field} {Tag_Field} {SF_Type} {Use_Z}...> {out_tin}`



- Multiple feature classes can be added at the same time.
- Before features can be added to a TIN, the TIN must be defined using the CreateTIN function.
- Features can be added to a newly created or existing TIN.



TIN Surface toolset

Contains tools to analyze the surface of a TIN dataset.

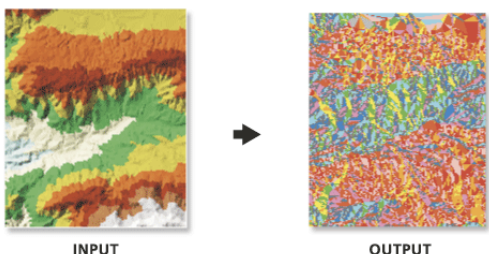
Interpolate Polygon To Multipatch: converts a polygon feature class to a multipatch feature class whose heights are based on a surface.

`InterpolatePolyToPatch <in_tin> <in_feature_class> <out_feature_class> {max_strip_size}
{z_factor}`

- Convert polygons to multipatches if you are having display problems with polygons conforming to a surface.
- The Maximum Triangle Strip Size value must be 3 or larger. The recommended range is between 128 and 2,048.

TIN Aspect: calculates aspect for the surface using each triangle in a TIN.

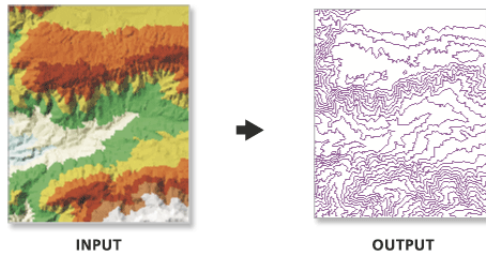
`TinAspect <in_tin> <out_feature_class> {class_breaks_table} {aspect_field}`



- Aspect is expressed in degrees.

TIN Contour: derives contour lines for a surface from a TIN.

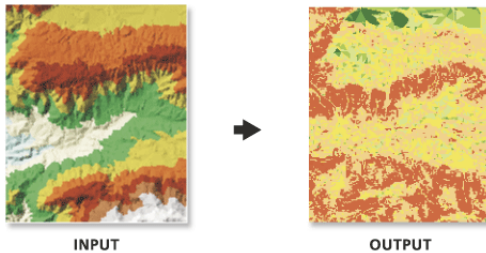
```
TinContour <in_tin> <out_feature_class> <interval> {base_contour} {contour_field}
{contour_field_precision} {index_interval} {index_interval_field} {z_factor}
```



- Use the interval and base contour options to tailor the extent and resolution of the output feature class.
- Use the out contour field as the height source for displaying the contours in 3D.

TIN Slope: calculates the slope of the surface as the maximum rate of change in elevation across each triangle.

```
TinSlope <in_tin> <out_feature_class> {PERCENT | DEGREE} {class_breaks_table}
{slope_field} {z_factor}
```



- Use {class_breaks_table} to constrain slope information into specific break intervals of the output feature class.
- Units are only honored when using a class breaks table.



Data Interoperability toolbox

Contains tools to import and export data with the ArcGIS Data Interoperability extension.

Quick Export: Converts one or more input feature classes or feature layers to any format supported by the ArcGIS Data Interoperability extension.

`QuickExport <input;input...> <output>`

- This tool is used either to export data from ArcGIS or as the final step in a model or script where the destination data is external to ArcGIS.
- During the export, no change to the data model is made; if this is desired, a custom data export tool should be created and used.

Quick Import: Converts data from any format supported by the ArcGIS Data Interoperability extension to a personal geodatabase.

`QuickImport <input> <output>`

- This tool is used either to bring data into the ArcGIS environment or as the beginning point in a model or script where data from outside ArcGIS will be processed.
- The feature classes generated depend on the input data. For instance, if you import two MapInfo MIF/MID® files, two feature classes will be created.
- This tool creates a new personal geodatabase and will not append to an existing one.
- The feature classes generated from the imported data can be accessed using the Select Data tool on the output staging personal geodatabase.
- As data is imported, no changes to the data model are made. To transform the data model during import, a custom data import tool should be created and used.



Geostatistical Analyst toolbox

Contains tools for exploratory spatial data analysis using the ArcGIS Geostatistical Analyst extension.

GA Layer To Grid: exports a geostatistical layer to an ArcInfo GRID.

```
GALayerToGrid <in_geostat_layer> <out_surface_grid> <cell_size> <points_per_block_horz>  
               <points_per_block_vert>
```

- The output grid will be created at the cell size specified by output cell size.
- Select the number of predictions for each cell in the horizontal and vertical directions for block interpolation.



Network Analyst toolbox

Contains tools for network analysis, building networks and creating and editing turns.



Analysis toolset

Contains tools used to perform analysis when using the ArcGIS Network Analyst extension.

Add Field To Analysis Layer: adds a field to a network analysis layer.

```
AddFieldToAnalysisLayer <in_network_analysis_layer> <sub_layer> <field_name> <LONG | TEXT  
| FLOAT | DOUBLE | SHORT | DATE | BLOB> {field_precision} {field_scale} {field_length}  
{field_alias} {NULLABLE | NON_NULLABLE}
```

- Fields can be added to any of the sublayers of the network analysis layers. These fields can be updated with values and mapped to various field mappings of the Add Locations tool to provide inputs to the network analysis layer.

Add Locations: adds network locations to a network analysis layer.

```
AddLocations <in_network_analysis_layer> <sub_layer> <in_table> <field_mappings>  
<search_tolerance> {sort_field} {source {snap type}; source {snap type}...}  
{MATCH_TO_CLOSEST | PRIORITY} {APPEND | CLEAR}
```

- Add Locations can be run repeatedly to append locations to the same sublayer. For example, if stops for a route layer come from two feature classes, Add Locations can be called twice by using the APPEND option.

Calculate Locations: calculates the network location fields for a point feature class.

```
CalculateLocations <in_point_features> <in_network_dataset> <search_tolerance> <source  
{snap type}; source {snap type}...> {MATCH_TO_CLOSEST | PRIORITY} {source_ID_field}  
{source_OID_field} {position_field} {side_field} {snap_X_field} {snap_Y_field} {distance_  
field}
```

- Calculate locations should be used on point data that will be used more than once as input to a network analysis layer. Once the locations have been computed, then the Add Locations tool allows the Use Network Location Fields instead of Geometry to be used to load the locations very quickly.

Directions: generates directions information for a network analysis layer with routes. The directions information is written to a file in either XML or text format.

```
Directions <in_network_analysis_layer> <XML | TEXT> <out_directions_file> <Feet | Yards |  
Miles | Meters | Kilometers> {REPORT_TIME | NO_REPORT_TIME} {time_attribute}
```

- Directions cannot be generated for analysis layer files (.lyr), only for analysis layers in the ArcMap table of contents or layers created in the same application session, script, or model as the Directions command is run.

Make Closest Facility Layer: makes a closest facility network analysis layer and sets its navigation properties.

```
MakeClosestFacilityLayer <in_network_dataset> <out_network_analysis_layer>  
<impedance_attribute> {TRAVEL_TO | TRAVEL_FROM} {default_cutoff} {default_number_  
facilities_to_find} {accumulate_attribute_name; accumulate_attribute_name...}  
{ALLOW_UTURNS | NO_UTURNS | ALLOW_DEAD_ENDS_ONLY} {restriction_attribute_name;  
restriction_attribute_name...} {USE_HIERARCHY | NO_HIERARCHY} {hierarchy_settings}  
{TRUE_LINES_WITH_MEASURES | STRAIGHT_LINES | NO_LINES}
```

- A closest facility layer can be made on a network currently added to the ArcMap table of contents, or you can browse to a network dataset on disk.
- Hierarchy options can be set only if the input analysis network has a hierarchy attribute.

Make OD Cost Matrix Layer: makes an origin and destination cost matrix layer and sets its navigation properties.

```
MakeODCostMatrixLayer <in_network_dataset> <out_network_analysis_layer>
<impedance_attribute> {default_cutoff} {default_number_destinations_to_find}
{accumulate_attribute_name; accumulate_attribute_name...} {ALLOW_UTURNS | NO_UTURNS
| ALLOW_DEAD_ENDS_ONLY} {restriction_attribute_name; restriction_attribute_name...}
{USE_HIERARCHY | NO_HIERARCHY} {hierarchy_settings} {STRAIGHT_LINE | NO_SHAPE}
```

- An OD cost matrix layer can be made on a network currently added to the ArcMap table of contents or on a network dataset on disk.
- Hierarchy options can be set only if the input analysis network has a hierarchy attribute.

Make Route Layer: makes a route network analysis layer and sets its navigation properties.

```
MakeRouteLayer <in_network_dataset> <out_network_analysis_layer> <impedance_attribute>
{USE_INPUT_ORDER | FIND_BEST_ORDER} {PRESERVE_BOTH | PRESERVE_NONE | PRESERVE_
FIRST | PRESERVE_LAST} {USE_TIMEWINDOWS | NO_TIMEWINDOWS} {accumulate_attribute_
name; accumulate_attribute_name...} {ALLOW_UTURNS | NO_UTURNS | ALLOW_DEAD_ENDS_ONLY}
{restriction_attribute_name; restriction_attribute_name...} {USE_HIERARCHY | NO_
HIERARCHY} {hierarchy_settings} {TRUE_LINES_WITH_MEASURES | STRAIGHT_LINES | NO_LINES}
```

- The Route layer can be made on a network currently added to the ArcMap table of contents or on a network dataset on disk.
- Hierarchy options can be set only if the input analysis network has a hierarchy attribute.

Make Service Area Layer: makes a service area network analysis layer and sets its navigation properties.

```
MakeServiceAreaLayer <in_network_dataset> <out_network_analysis_layer> <impedance_
attribute> {TRAVEL_FROM | TRAVEL_TO} {default_break_values} {SIMPLE_POLYGONS |
DETAILED_POLYGONS | NO_POLYS} {NO_MERGE | MERGE} {RINGS | DISKS} {NO_LINES | TRUE_LINES
| LINES_WITH_MEASURES} {OVERLAP | NON_OVERLAP} {NO_SPLIT | SPLIT} {excluded_source_
name; excluded_source_name...} {accumulate_attribute_name; accumulate_attribute_
name...} {ALLOW_UTURNS | NO_UTURNS | ALLOW_DEAD_ENDS_ONLY} {restriction_attribute_
name; restriction_attribute_name...}
```

- The service area layer can be made on a network currently added to the ArcMap table of contents or on a network dataset on disk.
- Hierarchy options can be set only if the input analysis network has a hierarchy attribute.

Solve: performs the analysis appropriate to the network analysis layer on which it is executed.

```
solve_na <in_network_analsis_layer> {SKIP | HALT}
```

- Be sure to specify all the parameters necessary to perform the analysis before running Solve.



Network Dataset toolset

Contains tools used to build network datasets.

Build Network: reconstructs the network connectivity and attribute information of a network dataset.

```
buildNetwork_na <in_network_dataset>
```



Turn Feature Class toolset

Contains tools used to build and edit turn data.

Create Turn Feature Class: creates a new turn feature class.

```
CreateTurnFeatureClass <out_location> <out_feature_class_name> {maximum_edges}  
  {in_network_dataset} {in_template_feature_class} {spatial_reference} {config_keyword}  
  {spatial_grid_1} {spatial_grid_2} {spatial_grid_3}
```

Increase Maximum Edges: increases the maximum number of edges in a turn feature class.

```
IncreaseMaximumEdges <in_turn_features> <maximum_edges>
```

Populate Alternate ID Fields: creates and populates additional fields on the turn feature classes in a network dataset that reference the edges by alternate IDs.

```
PopulateAlternateIDFields <in_network_dataset> <alternate_ID_field_name>
```

Turn Table To Turn Feature Class: converts an ArcView GIS turn table or ArcInfo Workstation coverage turn table to an ArcGIS turn feature class.

```
TurnTableToFeatureClass <in_turn_table> <reference_line_features> <out_feature_class_  
  name> {reference_nodes_table} {maximum_edges} {config_keyword} {spatial_grid_1}  
  {spatial_grid_2} {spatial_grid_3}
```

Update By Alternate ID Fields: updates all the edge references in each turn feature class in a network dataset using an alternate ID field.

```
UpdateByAlternateIDFields <in_network_dataset> <alternate_ID_field_name>
```

Update By Geometry: updates all the edge references in the turn table using the geometry of the feature.

```
UpdateByGeometry <in_turn_features>
```


Spatial Analyst toolbox

This toolbox provides tools for performing cell-based (raster) spatial analysis using the ArcGIS Spatial Analyst extension.

Conditional toolset

Contains tools to control the output values based on the conditions placed on the input values.

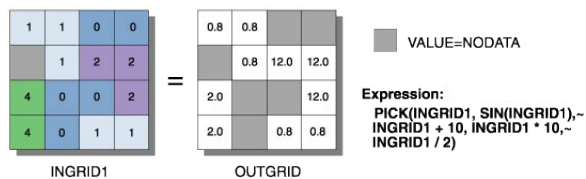
Con: performs a conditional if/else evaluation on each of the input cells of an input raster.

Con <in_conditional_raster> <in_true_raster_or_constant> <out_raster>
{in_false_raster_or_constant} {where_clause}

- If the evaluation of the expression is nonzero, it is treated as True.
- If no input false raster or constant is specified, NoData will be assigned to those cells that do not result in True from the expression.
- From the Command Line, the {where_clause} should be enclosed in quotes, for example, "Value > 5".

Pick: assigns output values using one of a list of rasters determined by the value of an input raster.

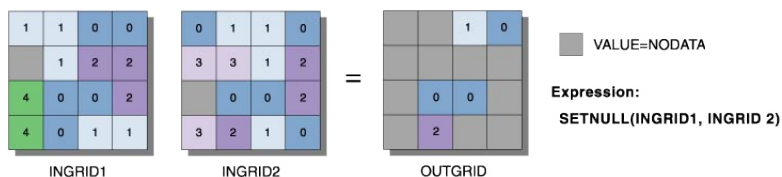
Pick <in_position_raster> <in_rasters_or_constants; in_rasters_or_constants...>
<out_raster>



- Pick is based on the order of the rasters in the input rasters or constant values. If the order of rasters changes, the results will change.
- The value of each cell of the input position raster determines which input raster (or constant value) will be used to obtain the output raster value.
- The input position raster should be a positive integer. If a cell value is 0 or negative, the result will be NoData. If the cell value is larger than the number of rasters in the input rasters or constant values, the result will be NoData.

Set Null: returns NoData if a conditional evaluation is true and returns the value specified by another raster if it is false, on a cell-by-cell basis.

SetNull <in_conditional_raster> <in_false_raster_or_constant> <out_raster>
{where_clause}



- If the evaluation of the where clause is true, the cell location on the output raster will be assigned NoData. If the evaluation is false, the output raster will be defined by the input false raster or constant value.
- If no where clause is specified, the output raster will have NoData wherever the conditional raster is greater than 0.



Density toolset

Contains tools used to calculate the spread of values over a surface.

Kernel Density: calculates a magnitude per unit area from point or polyline feature using a kernel function to fit a smoothly tapered surface to each point or polyline.

```
kernelDensity <in_features> <population_field> <out_raster> {cell_size} {search_radius}
{SQUARE_MAP_UNITS | SQUARE_MILES | SQUARE_KILOMETERS | ACRES | HECTARES | SQUARE_YARDS |
SQUARE_FEET | SQUARE_INCHES | SQUARE_METERS | SQUARE_CENTIMETERS | SQUARE_MILLIMETERS}
```

- Only the points or portions of a line that fall within the neighborhood are considered in calculating the density. If no points or line sections fall within the neighborhood of a particular cell, that cell is assigned NoData.
- Larger values of the radius parameter produce a smoother, more generalized density raster. Smaller values produce a raster that shows more detail.
- If the area unit scale factor units are small relative to the features (distance between points or length of line sections, depending on feature type), the output values may be small. To obtain larger values, select the area unit scale factor for larger units (for example, square kilometers versus square meters).

Line Density: calculates a magnitude per unit area from polyline features that fall within a radius around each cell.

```
LineDensity <in_polyline_features> <population_field> <out_raster> {cell_size}
{search_radius} {SQUARE_MAP_UNITS | SQUARE_MILES | SQUARE_KILOMETERS | ACRES | HECTARES |
SQUARE_YARDS | SQUARE_FEET | SQUARE_INCHES | SQUARE_METERS | SQUARE_CENTIMETERS |
SQUARE_MILLIMETERS}
```

- Only the portion of a line within the neighborhood is considered in calculating the density. If no lines fall within the neighborhood at a particular cell, that cell is assigned NoData.
- If the area unit scale factor units are small relative to the features (length of line sections), the output values may be small. To obtain larger values, use the area unit scale factor for larger units (for example, square kilometers versus square meters).
- Larger values of the radius parameter produce a smoother, more generalized density raster. Smaller values produce a raster that shows more detail.

Point Density: calculates a magnitude per unit area from point features that fall within a neighborhood around each cell.

```
PointDensity <in_point_features> <population_field> <out_raster> {cell_size}
{neighborhood} {SQUARE_MAP_UNITS | SQUARE_MILES | SQUARE_KILOMETERS | ACRES | HECTARES |
SQUARE_YARDS | SQUARE_FEET | SQUARE_INCHES | SQUARE_METERS | SQUARE_CENTIMETERS |
SQUARE_MILLIMETERS}
```

- Only the points that fall within the neighborhood are considered in calculating the density. If no points fall within the neighborhood at a particular cell, that cell is assigned NoData.
- Larger values of the radius parameter produce a smoother, more generalized density raster. Smaller values produce a raster that shows more detail.
- If the area unit scale factor units are small relative to the distance between the points, the output raster values may be small. To obtain larger values, use the area unit scale factor for larger units (for example, square kilometers versus square meters).



Distance toolset

Contains tools used to compute distance across a raster dataset with respect to cost or along a path.

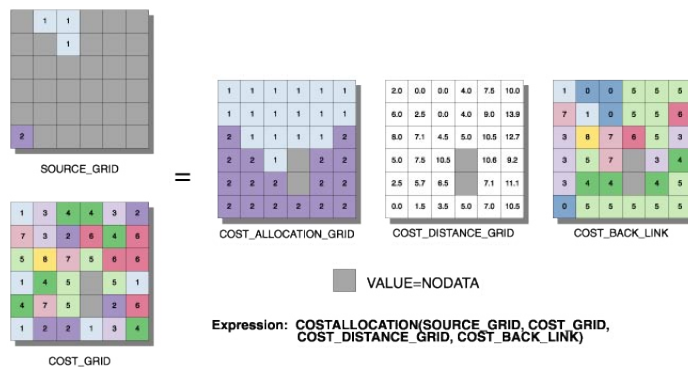
Corridor: computes the sum of two accumulative cost raster datasets.

`Corridor <in_distance_raster1> <in_distance_raster2> <out_raster>`

- Any two floating-point rasters can be used for the input (for example, an accumulative cost raster), but both inputs should be the result of a global cost function.
- The order of input is irrelevant.

Cost Allocation: identifies for each cell the zone of each source cell that could be reached with the least accumulative cost.

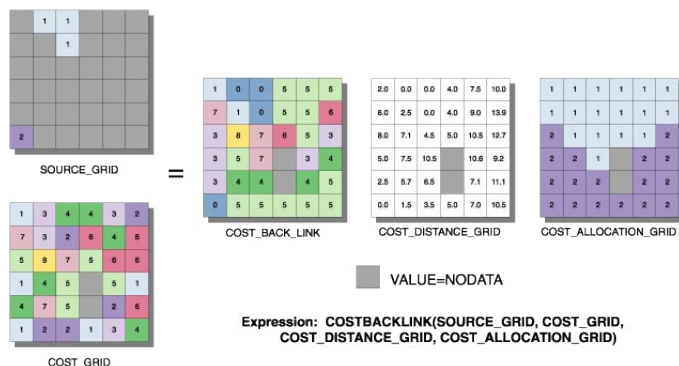
`CostAllocation <in_source_data> <in_cost_raster> <out_allocation_raster> {maximum_distance} {in_value_raster} {source_field} {out_distance_raster} {out_backlink_raster}`



- When the input source data is a raster, the set of source cells consists of all cells in the source raster that have valid values. Cells that have NoData values are not included in the source set. The value 0 is considered a legitimate source. A source raster can be conveniently created using the Extract By tools.
- Cell locations with NoData in the input cost raster act as barriers in the cost surface functions. Any cell location that is assigned NoData on the input cost surface will receive NoData on all outputs (cost allocation raster and optionally cost distance and cost back link rasters).
- The Input value raster is useful if the Input source data is a raster and was derived from a function or operator that results in either 1 or 0. These functions lose the original zone values on the input raster that are associated with these locations. The input value raster can either restore these values or allow for analysis on additional combinations of zone values within the source locations.
- The maximum distance is specified in the same cost units as those on the input cost raster.

Cost Back Link: defines the neighbor cell on the least accumulative cost path from a cell to a set of source cells.

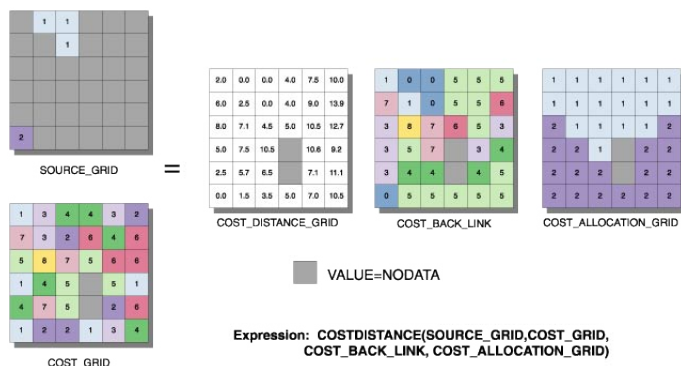
`CostBackLink <in_source_data> <in_cost_raster> <out_backlink_raster> {maximum_distance} {out_distance_raster}`



- When the input source data is a raster, the set of source cells consists of all cells in the source raster that have valid values. Cells that have NoData values are not included in the source set. The value 0 is considered a legitimate source. A source raster can be conveniently created using the Extract By tools.
- Cell locations with NoData in the input cost raster act as barriers in the cost surface functions. Any cell location that is assigned NoData on the input cost surface will receive NoData on all outputs (cost back link raster and optional cost distance and cost allocation rasters).
- The input value raster is useful if the Input source data is a raster and was derived from a function or operator that results in either 1 or 0. These functions lose the original zone values on the input raster that are associated with these locations. The input value raster can either restore these values or allow for analysis on additional combinations of zone values within the source locations.
- The maximum distance is specified in the same cost units as those on the input cost raster.

Cost Distance: calculates the least accumulative cost distance over a cost surface.

`CostDistance <in_source_data> <in_cost_raster> <out_distance_raster> {maximum_distance}
{out_backlink_raster}`

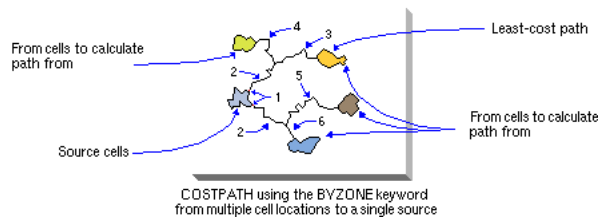


- When the input source data is a raster, the set of source cells consists of all cells in the source raster that have valid values. Cells that have NoData values are not included in the source set. The value 0 is considered a legitimate source. A source raster can be conveniently created using the Extract By tools.
- Cell locations with NoData in the Input cost raster act as barriers in the cost surface functions. Any cell location that is assigned NoData on the input cost surface will receive NoData on all outputs (cost distance raster and optional cost back link raster).
- The maximum distance is specified in the same cost units as those on the cost raster.
- The least cost distance or minimum accumulative cost distance of a cell to a set of source cells is the lower bound of the least cost distance from the cell to all source cells.

Cost Path: calculates the least cost paths from a source to a destination over a surface.

`CostPath <in_destination_data> <in_cost_distance_raster> <in_cost_backlink_raster>
<out_raster> {EACH_CELL | EACH_ZONE | BEST_SINGLE} {destination_field}`

- Produces an output raster that records the least cost path or paths from selected locations to the closest source cell defined within the accumulative cost surface, in terms of cost distance.
- When the input destination data is a raster, the set of destination cells consists of all cells in the Input raster or feature destination data that have valid values. Cells that have NoData values are not included in the source set. The value 0 is considered a legitimate destination. A destination raster may be conveniently created using the Extract By tools.
- One or more of the weighted cost functions (Cost Distance, Cost Back Link, or Cost Allocation) are generally required to be run prior to running Cost Path to create the Input cost distance raster and the input cost back link raster. These are mandatory input rasters to Cost Path.
- When multiple paths merge and follow the remaining distance back to a source on the same route, the segment where the two paths travel together is assigned the value 2. The merged portion of the path cannot be assigned the value of one of the paths, since the merged portion belongs to both routes.



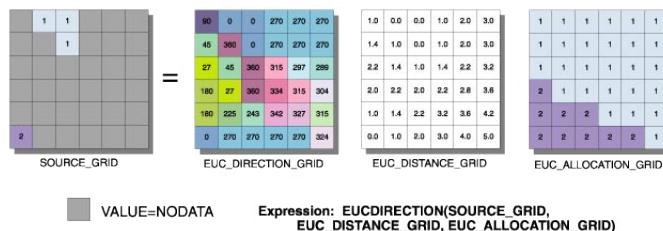
Euclidean Allocation: assigns each cell the value of the sources to which it is closest.

`EucAllocation <in_source_data> <out_allocation_raster> {maximum_distance} {in_value_raster} {cell_size} {source_field} {out_distance_raster} {out_direction_raster}`

- When the input source data is a raster, the set of source cells consists of all cells in the source raster that have valid values. Cells that have NoData values are not included in the source set. The value 0 is considered a legitimate source. A source raster may be conveniently created using the Extract By tools.
- The Euclidean functions are calculated from nonsource cells assigned a NoData value.
- The input value raster is useful if the input raster or feature source data is a raster derived from a function that results in either 1 or 0. These functions lose their original zone values that are associated with the source cell locations. The input value raster can either restore these values or allow for analysis on additional combinations of zone values within the source cells.
- The maximum distance is specified in the same map units as the input source data.

Euclidean Direction: computes for each cell the direction to the nearest source, measured in degrees.

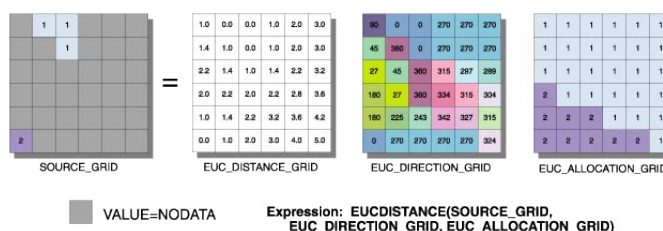
`EucDirection <in_source_data> <out_direction_raster> {maximum_distance} {cell_size} {out_distance_raster}`



- The output values are based on compass directions (90 to the east, 180 to the south, 270 to the west, and 360 to the north), with 0 being reserved for the source cells.
- When the input source data is a raster, the set of source cells consists of all cells in the source raster that have valid values. Cells that have NoData values are not included in the source set. The value 0 is considered a legitimate source. A source raster may be conveniently created using the Extract By tools.
- The Euclidean functions are calculated from nonsource cells assigned NoData values.
- The Maximum distance is specified in the same map units as the input source data.

Euclidean Distance: calculates for each cell the straight-line distance from each cell to the closest source.

`EucDistance <in_source_data> <out_distance_raster> {maximum_distance} {cell_size} {out_direction_raster}`



- When the input source data is a raster, the set of source cells consists of all cells in the source raster that have valid values. It must be an integer raster. Cells that have NoData values are not included in the source set. The value 0 is considered a legitimate source. A source raster may be conveniently created using the Extract By tools.
- The Euclidean functions are calculated from nonsource cells assigned NoData.
- The Maximum distance is specified in the same map units as the input source data.

Path Distance: calculates for each cell the least accumulative cost distance over a cost surface from a source cell or a set of source cells while accounting for surface distance and horizontal and vertical cost factors.

```
PathDistance <in_source_data> <out_distance_raster> {in_cost_raster} {in_surface_raster}
{in_horizontal_raster} {horizontal_factor} {in_vertical_raster} {vertical_factor}
{maximum_distance} {out_backlink_raster}
```

- When the input source data is a raster, the set of source cells consists of all cells in the source raster that have valid values. Cells that have NoData values are not included in the source set. The value 0 is considered a legitimate source. A source raster may be conveniently created using the Extract By tools.
- Cells with NoData act as barriers in the Path Distance functions. The cost distance for cells behind NoData values is calculated by the accumulative cost necessary to move around the NoData barrier. Any cell location that is assigned NoData on any one of the input rasters will receive NoData on all output rasters.
- The maximum distance is specified in the same cost units as those on the input surface raster.
- The least-cost-path distance or minimum accumulative cost distance of a cell from a set of source locations is the lower bound of the least-cost-path distance to the cell from all source locations.

Path Distance Allocation: calculates for each cell its nearest source based on the least accumulative cost over a cost surface while accounting for surface distance, horizontal cost factors, and vertical cost factors.

```
PathAllocation <in_source_data> <out_allocation_raster> {in_cost_raster}
{in_surface_raster} {in_horizontal_raster} {horizontal_factor} {in_vertical_raster}
{vertical_factor} {maximum_distance} {in_value_raster} {source_field}
{out_distance_raster} {out_backlink_raster}
```

- When the input source data is a raster, the set of source cells consists of all cells in the source raster that have valid values. Cells that have NoData values are not included in the source set. The value 0 is considered a legitimate source. A source raster may be conveniently created using the Extract By tools.
- Cells with NoData act as barriers in the Path Distance functions. The cost distance for cells behind NoData values is calculated by the accumulative cost necessary to move around the NoData barrier. Any cell location that is assigned NoData on any one of the input rasters will receive NoData on all output rasters.
- The maximum distance is specified in the same cost units as those on the input surface raster.
- The least-cost-path distance or minimum-accumulative-cost distance of a cell from a set of source locations is the lower bound of the least-cost-path distance to the cell from all source locations.

Path Distance Back Link: defines the neighbor that is the next cell on the least accumulative cost path to the nearest source while accounting for surface distance, horizontal cost factors, and vertical cost factors.

```
PathBackLink <in_source_data> <out_backlink_raster> {in_cost_raster} {in_surface_raster}
{in_horizontal_raster} {horizontal_factor} {in_vertical_raster} {vertical_factor}
{maximum_distance} {out_distance_raster}
```

- When the Input source data is a raster, the set of source cells consists of all cells in the source raster that have valid values. Cells that have NoData values are not included in the source set. The value 0 is considered a legitimate source. A source raster may be conveniently created using the Extract By tools.

- Cells with NoData act as barriers in the Path Distance functions. The cost distance for cells behind NoData values is calculated by the accumulative cost necessary to move around the NoData barrier. Any cell location that is assigned NoData on any one of the input rasters will receive NoData on all output rasters.
- The maximum distance is specified in the same cost units as those on the input surface raster.
- The least-cost-path distance or minimum accumulative cost distance of a cell from a set of source locations is the lower bound of the least-cost-path distance to the cell from all source locations.

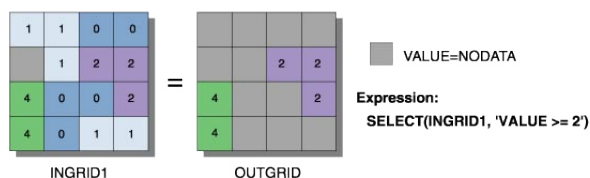


Extraction toolset

Contains tools to extract a subset of cells either by the attributes or spatial location of each cell.

Extract By Attributes: extracts the cells of a raster dataset based on a logical query.

`ExtractByAttributes <in_raster> <where_clause> <out_raster>`



- If the WHERE clause evaluates to true, the original input's value is returned for the cell location.
- If the WHERE clause evaluates to false, the cell location is assigned NoData.
- If the input raster is floating point, the query must reference Value. For example: `value > 10`
- If an item other than value of input raster is specified in the WHERE clause, the original input's value is returned for the cell location.

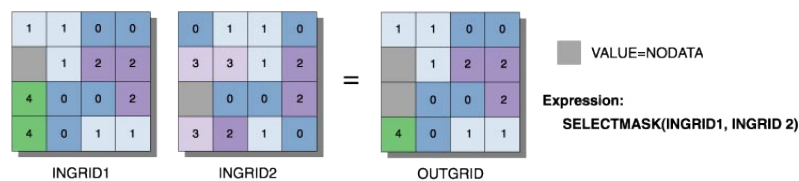
Extract By Circle: extracts the cell values of a raster dataset based on the boundaries of a circle.

`ExtractByCircle <in_raster> <center_point> <radius> <out_raster> {INSIDE | OUTSIDE}`

- The center of the cell is used to determine whether a cell is within or outside a circle. If the center is within the arc of the circle, the cell is considered fully inside even if portions of the cell fall outside the circle.
- Cell locations that are not selected are assigned a value of NoData.

Extract By Mask: extracts the raster dataset cell values based on a mask dataset.

`ExtractByMask <in_raster> <in_mask_data> <out_raster>`



- Where input raster or feature mask data is raster, the values for non-NoData input cell locations are copied to the output raster. Tools that can create the mask raster include Con, Test, and the Extraction group.
- Extract By Mask is similar to setting the Mask option in the Environment, except that the mask used in Extract By Mask is only used on the immediate instance, while a mask set in the environment is applied to all tools until it is changed or disabled.

Extract By Points: extracts the cells of a raster dataset based on a set of points.

`ExtractByPoints <in_raster> <points;points...> <out_raster> {INSIDE | OUTSIDE}`

- Cell locations that are not selected are assigned NoData.

Extract By Polygon: extracts the cells of a raster dataset based on the boundaries within a polygon feature.

`ExtractByPolygon <in_raster> <polygon;polygon...> <out_raster> {INSIDE | OUTSIDE}`

- The center of the cell is used to determine whether a cell is within or outside a polygon. If the center is within the arcs of the polygon, the cell is considered fully inside even if portions of the cell fall outside the polygon.
- The polygon has a limit of 1,000 vertices. Polygon vertices must be entered in a clockwise order. The first and last vertex must be the same to close the polygon. The arcs of the polygon can cross one another, but convoluted polygons are not recommended.
- Cell locations that are not selected are assigned values of NoData.

Extract By Rectangle: extracts the cells of a raster dataset based on the boundaries of a rectangle.

`ExtractByRectangle <in_raster> <rectangle> <out_raster> {INSIDE | OUTSIDE}`

- The center of the cell is used to determine whether a cell is within or outside a rectangle. If the center is within the outline of a rectangle, the cell is considered fully inside even if portions of the cell fall outside the rectangle.
- Cell locations that are not selected are assigned values of NoData.

Extract Values To Points: extracts the cell values from a raster at the locations of points in a feature class.

`ExtractValuesToPoints <in_point_features> <in_raster> <out_point_features> {NONE | INTERPOLATE} {VALUE_ONLY | ALL}`

- The interpolation option determines how the values will be obtained from the raster. The default option is to use the value at the center of the cell being sampled. The interpolation option will use bilinear interpolation to interpolate a value for the cell center.
- The Extract Values To Points tool can be used with a floating-point input raster. In this case, the resulting output point dataset will only contain attributes from the input feature data and the value of the cell, as determined by the interpolation option.
- The output shapefile will have at least the RASTERVALU field added. All other items will be appended after the RASTERVALU field.
- For the RASTERVALU field of the attribute table, NoData cells in the value raster will be given a value of -9999.

Sample: writes a sample of cell values from a group of rasters to a table.

`Sample <in_rasters;in_rasters...> <in_location_data> <out_table> {NEAREST | BILINEAR | CUBIC}`

- When the input location raster or point features are rasters, the set of location cells consists of all cells in the raster that have a value of 0 or greater. Cells that have NoData values are not included in the location set. A location raster may be conveniently created using the Extract By tools.
- If the input location raster or point features are rasters, NoData cells will be listed in the output table as -9999.

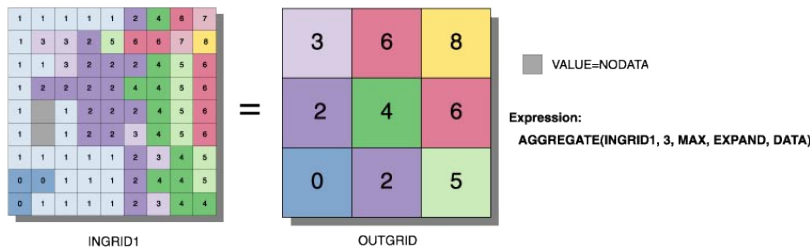


Generalization toolset

Contains tools to remove or reduce erroneous or irrelevant data within a raster through aggregation, edge smoothing, intelligent noise removal, and so forth.

Aggregate: generates a reduced resolution raster dataset.

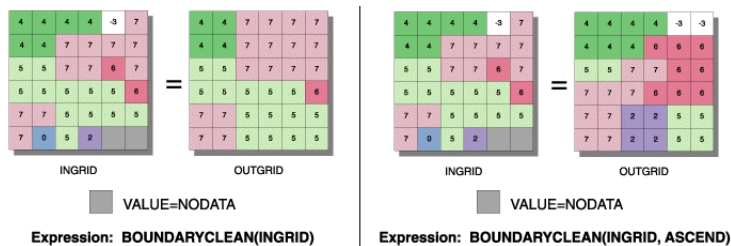
`Aggregate <in_raster> <out_raster> <cell_factor> {SUM | MAXIMUM | MEAN | MEDIAN | MINIMUM} {EXPAND | TRUNCATE} {DATA | NODATA}`



- The geoprocessing analysis environment (extent, cell size, and mask) is recognized by the Aggregate function. To determine the output raster's resolution when an integer cell size has been specified, multiply the cell resolution of the analysis environment by the input cell factor parameter. If the cell size for the analysis environment is set to the minimum or maximum of the inputs, the resolution of the output raster will be the product of the input raster's resolution multiplied by the specified cell factor.

Boundary Clean: smooths the boundary between zones by expanding and shrinking the boundary.

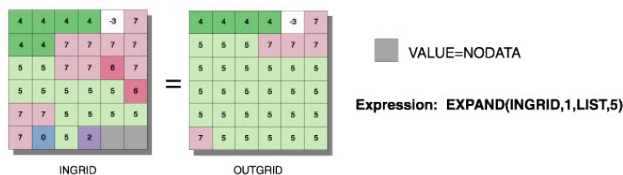
BoundaryClean <in_raster> <out_raster> {NO_SORT | DESCEND | ASCEND} {TWO_WAY | ONE_WAY}



- All regions of less than three cells in the x or y direction will be changed.
- In the first pass, for any processing cell in the expanded raster that has a neighbor of the original value of the processing cell, the original value of the processing cell will be recovered. In the second pass of TWO_WAY, any cell in the expanded raster that is not completely surrounded by eight cells of the same value will recover its original value.
- In the ONE_WAY or first pass of the TWO_WAY, cells of NoData have the lowest priority. In the second pass of the TWO_WAY, cells of NoData have the highest priority.

Expand: expands the selected zones of a raster dataset by a specified number of cells.

Expand <in_raster> <out_raster> <number_cells> <zone_values;zone_values...>



- The zone values must be integers. They can be in any order.

Majority Filter: replaces cell values within a raster dataset based on the majority of their contiguous neighboring cells.

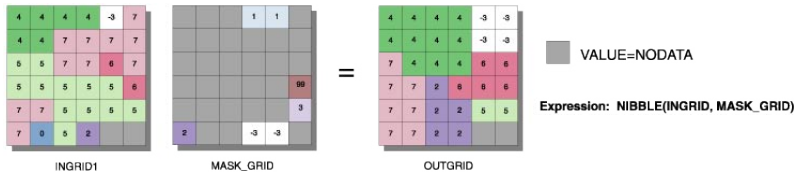
MajorityFilter <in_raster> <out_raster> {FOUR | EIGHT} {MAJORITY | HALF}



- The use of FOUR will retain the corners of rectangular regions. The use of EIGHT will smooth the corners of rectangular regions.
- Contiguous is defined as sharing an edge for a kernel of EIGHT and as sharing a corner for a kernel of FOUR.
- If the keyword HALF is specified and two values occur as equal halves, a replacement will not occur if the value of the processing cell is the same as one of the halves. HALF allows more extensive filtering than MAJORITY.
- Although the contiguity criterion is the same for edge and corner raster cells, they obey different MAJORITY and HALF rules. Using a kernel of FOUR, an edge or corner cell always requires two matching neighbors before replacement will occur. With a kernel of EIGHT, a corner cell must have all neighbors of the same value before it is changed, while an edge cell requires three contiguous neighbors, including one along the edge, before any change will occur.

Nibble: replaces cell values in a raster dataset corresponding to a mask with the values of the nearest neighbors.

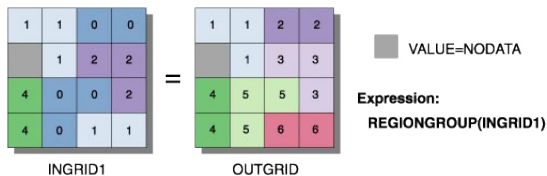
`nibble <in_raster> <in_mask_raster> <out_raster> {ALL_VALUES | DATA_ONLY}`



- Cells in the input raster containing NoData are not nibbled. To nibble away NoData, first convert it to another value.

Region Group: records for each cell of a raster the identity of the connected region to which it belongs. A unique number is assigned to each region.

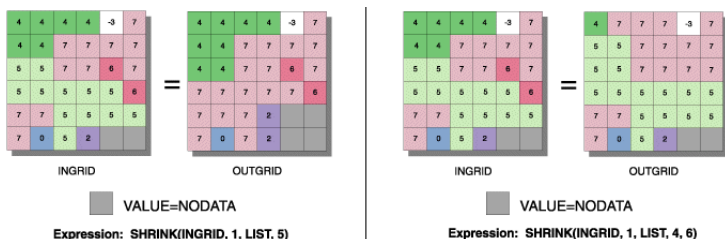
`RegionGroup <in_raster> <out_raster> {FOUR | EIGHT} {WITHIN | CROSS} {ADD_LINK | NO_LINK} {excluded_value}`



- The first region scanned (scan moves from left to right, top to bottom) receives the value 1, the second 2, and so forth, until all regions are assigned a value. The values assigned to the output zones are based on when they are encountered in the scanning process. The scanning process cannot be controlled by the user.
- By default, the Add Link option is True. This will create an item called LINK in the attribute table of the output raster, which retains the original value for each cell from the input raster.

Shrink: shrinks the selected zones by a specified number of cells.

`shrink <in_raster> <out_raster> <number_cells> <zone_values;zone_values...>`



- When two adjacent regions are part of the selected set to shrink, there is no change at the boundary between them.

- NoData has the same priority as any valid value to invade areas vacated by shrinking selected values. Therefore, if a selected value is adjacent to NoData, it may become NoData after shrinking.

Thin: thins rasterized linear features in a raster dataset.

Thin <in_raster> <out_raster> {ZERO | NODATA} {NO_FILTER | FILTER} {ROUND | SHARP} {maximum_thickness}

- The FILTER option uses the same filtering algorithm as Boundary Clean to remove short linear features extending from the major branch. It may also remove features narrower than three cells.
- Specifying the maximum thickness of input linear features is essential for thinning rasters where the thickness of linear features may well exceed or stay well below the default maximum thickness value. The best results can be expected when the maximum thickness fits the thickest linear features to be thinned.



Groundwater toolset

Contains tools used to measure hydrodynamic movement within or along a surface.

Darcy Flow: calculates the groundwater volume balance residual and other outputs for steady flow in an aquifer.

DarcyFlow <in_head_raster> <in_porosity_raster> <in_thickness_raster>
<in_transmissivity_raster> <out_volume_raster> {out_direction_raster}
{out_magnitude_raster}

- The only differences between Darcy Flow and Darcy Velocity are:
 - Darcy Flow produces an output volume raster. Darcy Velocity does not.
 - Darcy Velocity outputs only direction and magnitude rasters as required output. Darcy Flow produces these outputs optionally.
- The direction of the velocity vector is recorded in compass coordinates (degrees clockwise from north), the magnitude in units of length over time.
- No particular system of units is specified by this function. All data should be consistent, using the same unit for time (seconds, days, years) and length (feet, meters) for all data.
- However the head elevation raster is obtained, the head must be consistent with the transmissivity raster. That is, the head must reflect the flow through the transmissivity field. It is not sufficient to use values obtained by measurement and testing in the field—the gridded values must be analyzed for consistency with the aid of a proper porous medium flow program. Consistency implies that the heads would actually be produced by the modeled transmissivity field. Since the true and modeled transmissivity fields often differ in practice, the true and modeled head fields differ as well. Check the heads for consistency by examining the residual raster produced by Darcy Flow. The residual will reflect the consistency of the dataset. Any analysis using Darcy Flow on inconsistent datasets will produce meaningless results.
- The effective porosity field, a physical property of the aquifer, will generally be estimated from geological data. It is defined as the volume of void space that contributes to fluid flow divided by the entire volume. Porosity is expressed as a number between 0 and 1, with typical values of approximately 0.35, and is dimensionless. A value of effective porosity of 0.35 means that 35 percent of the volume of the porous medium contributes to fluid flow. The remaining 65 percent, consisting of solid matrix and unconnected pores, does not contribute to fluid flow.
- The saturated thickness, measured in units of length, is interpreted from geological information. For a confined aquifer, this measure is the thickness of the formation between the upper and lower confining layers. For an unconfined aquifer, the saturated thickness is the distance between the water table and the lower confining layer.

Darcy Velocity: calculates the groundwater seepage velocity vector (direction and magnitude) for steady flow in an aquifer.

`DarcyVelocity <in_head_raster> <in_porosity_raster> <in_thickness_raster>
<in_transmissivity_raster> <out_direction_raster> <out_magnitude_raster>`

- The only differences between Darcy Flow and Darcy Velocity are:
 - Darcy Flow produces an output volume raster. Darcy Velocity does not.
 - Darcy Velocity outputs only direction and magnitude rasters as required output. Darcy Flow produces these outputs optionally.
- The direction of the velocity vector is recorded in compass coordinates (degrees clockwise from north), the magnitude in units of length over time.
- No particular system of units is specified by this function. All data should be consistent, using the same unit for time (seconds, days, years) and length (feet, meters) for all data.
- However the head elevation raster is obtained, the head must be consistent with the transmissivity raster. That is, the head must reflect the flow through the transmissivity field. It is not sufficient to use values obtained by measurement and testing in the field—the rasterized values must be analyzed for consistency with the aid of a proper porous medium flow program. Consistency implies that the heads would actually be produced by the modeled transmissivity field. Since the true and modeled transmissivity fields often differ in practice, the true and modeled head fields differ as well. Check the heads for consistency by examining the residual raster produced by Darcy Flow. The residual will reflect the consistency of the dataset. Any analysis using Darcy Velocity on inconsistent datasets will produce meaningless results.
- The effective porosity field, a physical property of the aquifer, will generally be estimated from geological data. It is defined as the volume of void space that contributes to fluid flow divided by the entire volume. Porosity is expressed as a number between 0 and 1, with typical values of approximately 0.35, and is dimensionless. A value of effective porosity of 0.35 means that 35 percent of the volume of the porous medium contributes to fluid flow. The remaining 65 percent, consisting of solid matrix and unconnected pores, does not contribute to fluid flow.
- The saturated thickness, measured in units of length, is interpreted from geological information. For a confined aquifer, this measure is the thickness of the formation between the upper and lower confining layers. For an unconfined aquifer, the saturated thickness is the distance between the water table and the lower confining layer.

Particle Track: calculates the path of a particle through a velocity field.

`ParticleTrack <in_direction_raster> <in_magnitude_raster> <source_point> <out_track_file>
{step_length} {tracking_time} {out_track_polyline_features}`

- The input direction and magnitude rasters should be from the same run of Darcy Flow.
- The path file generated by this function is an ASCII text file containing information about position, local velocity direction and magnitude, and cumulative length and time of travel along the path. This file is used for input by Porous Puff. The format of this file is as follows:

Time	X	Y	Length	Flow Direction	Flow Magnitude
0.000000000	0.000000000	482.8400000	0.000000000	90.00000000	0.04418909563
113.1648712	4.999804443	482.7957786	5.000000000	91.01366126	0.04418332249
226.2741353	9.998043277	482.6630814	10.00000000	92.02765240	0.04421519432
339.3574334	14.99315255	482.4419855	15.00000000	93.04094157	0.04421519432
452.3447720	19.98356700	482.1325285	20.00000000	94.05521317	0.04425274599
565.2657591	24.96772671	481.7348453	25.00000000	95.06807622	0.04427874865
678.0514031	29.94406931	481.2490323	30.00000000	96.08254679	0.04433188322
790.7309576	34.91104149	480.6752838	35.00000000	97.09488082	0.04437362239

- No particular system of units is specified by Particle Track. It is important that all data supplied be in a consistent set of units, using the same unit for time (seconds, days, years) and length (feet, meters) for all data.
- The two outputs from Particle Track are (1) a particle track ASCII file using the name specified as output particle track file and (2) a polyline feature class (optional).

Porous Puff: calculates the hydrodynamic dispersion of an instantaneous point release of a constituent as it is advected along the flow path.

PorousPuff <in_track_file> <in_porosity_raster> <in_thickness_raster> <out_raster> <mass> {dispersion_time} {longitudinal_dispersivity} {dispersivity_ratio} {retardation_factor} {decay_coefficient}

- No particular system of units is specified by this function. It is important that all data supplied be consistent, that is, using the same unit for time (seconds, days, years), length (feet, meters), and mass (kilograms, slugs) for all data.
- The effective porosity field, a physical property of the aquifer, will generally be estimated from geological data. It is defined as the volume of void space that contributes to fluid flow divided by the entire volume. Porosity is expressed as a number between 0.0 and 1.0, with typical values of approximately 0.35, and is dimensionless. A value of effective porosity of 0.35 means that 35 percent of the volume of the porous medium contributes to fluid flow. The remaining 65 percent, consisting of solid matrix and unconnected pores, does not contribute to fluid flow.
- The saturated thickness, measured in units of length, is interpreted from geological information. For a confined aquifer, this measure is the thickness of the formation between the upper and lower confining layers. For an unconfined aquifer, the saturated thickness is the distance between the water table and the lower confining layer.
- The decay coefficient λ is related to the half-life $T_{1/2}$ as:
$$\lambda = \frac{\ln 2}{T_{1/2}}$$

For example, the half-life of Carbon-14 is 5,730 years, so its decay coefficient is $\ln 2 = 0.693$. So the equation becomes $0.693/5730 = 1.21 \times 10^{-4}/\text{year}$. A stable constituent has a decay coefficient of zero, corresponding to an infinite half-life. Half-lives of radioisotopes are available from several sources, including the CRC Handbook of Chemistry and Physics.



Hydrology toolset

Contains tools providing hydrology functions to simulate the flow of water over an elevation surface and creates either a stream network or a watershed.

Basin: creates a raster dataset delineating all drainage basins.

Basin <in_flow_direction_raster> <out_raster>

- Best results will be obtained from basin if the force option was used when creating the flow direction raster.
- All cells in the raster will belong to a basin, even if that basin is only one cell.

Fill: fills sinks in a surface raster to remove small imperfections in the data.

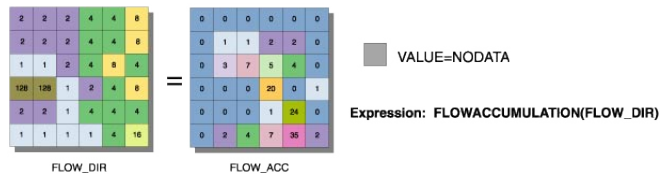
Fill <in_surface_raster> <out_surface_raster> {z_limit}

- A sink is a cell with an undefined drainage direction; no cells surrounding it are lower. The pour point is the boundary cell with the lowest elevation for the contributing area of a sink. If the sink were filled with water, this is the point where water would pour out.
- All sinks that are less than the z-limit lower than their lowest adjacent neighbor will be filled to the height of their pour points.

- The Sink tool can be used to find the number of sinks and help identify their depth. Knowing the depth of the sinks can help in determining an appropriate z-limit for Fill.
- Fill can also be used to remove peaks. A peak is a cell where no adjacent cells are higher. To remove peaks, the input surface raster must be inverted.

Flow Accumulation: creates a raster dataset of accumulated flow to each cell by accumulating the weight for all cells that flow into each downslope cell.

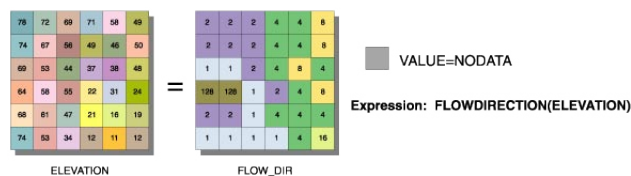
`FlowAccumulation <in_flow_direction_raster> <out_accumulation_raster> {in_weight_raster}`



- The result of Flow Accumulation is a raster of accumulated flow to each cell, as determined by accumulating the weight for all cells that flow into each downslope cell.
- The accumulated flow is based on the number of cells flowing into each cell in the output raster. The current processing cell is not considered in this accumulation.
- Output cells with a high flow accumulation are areas of concentrated flow and may be used to identify stream channels.
- Output cells with a flow accumulation of zero are local topographic highs and may be used to identify ridges.

Flow Direction: creates a raster dataset of flow direction from each cell to its steepest downslope neighbor.

`FlowDirection <in_surface_raster> <out_flow_direction_raster> {NORMAL | FORCE} {out_drop_raster}`



- The output of the Flow Direction tool is an integer raster whose values range from 1 to 255. The values for each direction from the center are:

32	64	128
16		1
8	4	2

For example, if the direction of steepest drop is to the left of the current processing cell, its flow direction would be coded as 16.

- If a cell is lower than its eight neighbors, that cell is given the value of its lowest neighbor and flow is defined toward this cell.
- If a cell has the same change in z-value in multiple directions and that cell is part of a sink, the flow direction is referred to as undefined. In such cases, the value for that cell in the output flow direction raster will be the sum of those directions.

Flow Length: calculates upstream or downstream distance or weighted distance along a flow path for each cell.

`FlowLength <in_flow_direction_raster> <out_raster> {DOWNSTREAM | UPSTREAM} {in_weight_raster}`

- The value type for the <out_raster> is floating point.

Sink: creates a raster dataset identifying all sinks or areas of internal drainage.

`sink <in_flow_direction_raster> <out_raster>`

- The output of the Sink function is an integer raster with each sink being assigned a unique value. Sinks are numbered between 1 and the number of sinks.

Snap Pour Point: snaps pour points to the cell of highest flow accumulation within a specified distance.

`SnapPourPoint <in_pour_point_data> <in_accumulation_raster> <out_raster> <snap_distance> {pour_point_field}`

- The Snap Pour Point tool is used to ensure the selection of points of high-accumulated flow when delineating drainage basins using the Watershed tool. Snap Pour Point will search within a snap distance around the specified pour points for the cell of highest accumulated flow and move the pour point to that location.
- The output is an integer raster where the original pour point locations have been snapped to locations of higher accumulated flow.

Stream Link: assigns unique values to sections of a raster linear network between intersections.

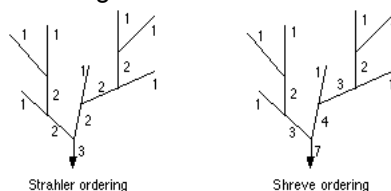
`StreamLink <in_stream_raster> <in_flow_direction_raster> <out_raster>`

- The input stream raster can be created by thresholding the results of Flow Accumulation.
- The stream raster linear network should be represented as values greater than or equal to 1 on a background of NoData.

Stream Order: assigns a numeric order to segments of a raster dataset representing branches of a linear network.

`StreamOrder <in_stream_raster> <in_flow_direction_raster> <out_raster> {STRAHLER | SHREVE}`

- The input stream raster linear network should be represented as values greater than or equal to 1 on a background of NoData.
- The results of Flow Accumulation can be used to create a raster stream network by applying a threshold value to select cells with a high accumulated flow. For example, the cells that have more than 100 cells flowing into them are used to define the stream network. Use Con or Set Null to create a stream network raster where flow accumulation values of 100 or greater go to 1, and the remainder are put to the background (NoData). The resulting stream network can be used in Stream Link and Stream To Feature.
- In the STRAHLER order method, all links with no tributaries are assigned an order of 1 and are referred to as first order. When two first-order links intersect, the downslope link is assigned an order of 2. When two second-order links intersect, the downslope link is assigned an order of 3, and so on. Only when two links of the same order intersect will the order increase. This is the most common method of ordering.



The output from the Stream Order tool is an integer raster.

Stream To Feature: converts a raster dataset representing a linear network to features representing the linear network.

`StreamToFeature <in_stream_raster> <in_flow_direction_raster> <out_polyline_features> {SIMPLIFY | NO_SIMPLIFY}`

- The input stream raster linear network should be represented as values greater than or equal to 1 on a background of NoData.
- The results of Flow Accumulation can be used to create a raster stream network by applying a threshold value to select cells with a high accumulated flow. For example, the cells that have more than 100 cells flowing into them are used to define the stream network. Use Con or Set Null to create a stream network raster where Flow Accumulation values of 100 or greater go to 1, and the remainder are put to the background (NoData). The resulting stream network can be used in Stream Link and Stream To Feature.
- There should be contiguous features with the same value, such as the results of Stream Order or Stream Link. Stream To Feature should not be used on a raster where there are few adjacent cells of the same value.
- The arcs of the output shapefile will point downstream.

Watershed: determines the contributing area above a set of cells in a raster dataset.

`watershed <in_flow_direction_raster> <in_pour_point_data> <out_raster> {pour_point_field}`

- The value of each watershed will be taken from the value of the source in the input raster or feature pour point data. When the pour point is a raster dataset, the cell values will be used. When the pour point is a point feature dataset, the values will come from the specified field.
- Better results will be obtained if the Snap Pour Point tool is used beforehand to help locate the pour points to cells of high-accumulated flow.



Interpolation toolset

Contains tools to create a raster surface from point features.

IDW: interpolates a surface from points using an inverse distance weighted (IDW) technique.

`IDW <in_point_features> <z_field> <out_raster> {cell_size} {power} {search_radius} {in_barrier_polyline_features}`

- The barriers option is used to specify the location of linear features known to interrupt the surface continuity. These features do not have z-values. Cliffs, faults, or embankments are typical examples of barriers. Barriers limit the selected set of the input sample points used to interpolate output z-values to those samples on the same side of the barrier as the current processing cell.
- The output value for a cell using IDW is limited to the range of the values used to interpolate. Because IDW is a weighted distance average, the average cannot be greater than the highest or less than the lowest input. Therefore, it cannot create ridges or valleys if these extremes have not already been sampled (Watson and Philip, 1985).
- The best results from IDW are obtained when sampling is sufficiently dense with regard to the local variation you are attempting to simulate. If the sampling of input points is sparse or uneven, the results may not sufficiently represent the desired surface (Watson and Philip, 1985).

Krige: interpolates a raster dataset from a set of points using kriging.

`kriging <in_point_features> <z_field> <out_surface_raster> <semivariogram_props> {cell_size} {search_radius} {out_variance_prediction_raster}`

- The universal kriging types (linear with linear drift and linear with quadratic drift) assume that there is a structural component present and that the local trend varies from one location to another.
- The advanced parameters allow control of the semivariogram used for Kriging. A default value for Lag size is initially set to the default output cell size. For major range, partial sill, and nugget, a default value will be calculated internally if nothing is specified.
- Low values within the output variance of prediction raster indicate a high degree of confidence in the predicted value. High values may indicate a need for more data points.

Natural Neighbor: interpolates a surface from points using a natural neighbor technique.

`NaturalNeighbor <in_point_features> <z_field> <out_raster> {cell_size}`

- The Natural Neighbor tool can efficiently handle large numbers of input points. Other interpolators may have difficulty with large point datasets.

Spline: interpolates a surface from points using a minimum curvature spline technique.

`Spline <in_point_features> <z_field> <out_raster> {cell_size}
{REGULARIZED | TENSION} {weight} {number_points}`

- The resulting smooth surface from Spline passes exactly through the input points.
- The REGULARIZED option of Spline usually produces smoother surfaces than those created with the TENSION option.
- For the REGULARIZED option, higher values used for the Weight parameter produce smoother surfaces. The values entered for this parameter must be equal to or greater than zero. Typical values used are 0, 0.001, 0.01, 0.1, and 0.5. The weight is the square of the parameter, referred to in the literature as tau (τ).
- For the TENSION option, higher values entered for the Weight parameter result in somewhat coarser surfaces but with surfaces that closely conform to the control points. The values entered must be equal to or greater than zero. Typical values are 0, 1, 5, and 10. The weight is the square of the parameter, referred to in the literature as phi (Φ).
- The greater the value of Number of Points, the smoother the surface of the output raster.

Topo To Raster: generates a hydrologically correct raster dataset of elevation.

`TopoToRaster <feature_layer{Field} {Type};feature_layer{Field} {Type}...>
<out_surface_raster> {cell_size} {extent} {Margin} {minimum_z_value} {maximum_z_value} {ENFORCE | NO_ENFORCE | ENFORCE_WITH_SINK} {CONTOUR | SPOT} {maximum_iterations} {roughness_penalty} {discrete_error_factor} {vertical_standard_error} {tolerance_1} {tolerance_2} {out_stream_features} {out_sink_features} {out_diagnostic_file} {out_parameter_file}`

- Topo To Raster will only use four input data points for the interpolation of each output cell. All additional points are ignored. If too many points are encountered by the algorithm, an error may occur indicating the point dataset has too many points. The maximum number of points that can be used is $nrows * ncols$, where $nrows$ is the number of rows in the output raster and $ncols$ is the number of columns.
- Stream data always takes priority over point or contour data; therefore, elevation data points that conflict with descent down each stream are ignored. Stream data is a powerful way of adding topographic information to the interpolation, further ensuring the quality of the output DEM.
- Some typical values for the Tolerance 1 and Tolerance 2 settings are:
 - For point data at 1:100,000 scale, use 5.0 and 200.0.
 - For less dense point data at up to 1:500,000 scale, use 10.0 and 400.0.
 - For contour data with contour spacing of 10, use 5.0 and 100.0.

Topo To Raster By File: generates a hydrologically correct raster dataset of elevation.

`TopoToRasterByFile <in_parameter_file> <out_surface_raster> {out_stream_features} {out_sink_features}`

- The parameter file is structured with the input datasets listed first, followed by the various parameter settings, then the output options.

The input data identifies the input datasets and, where applicable, fields. There are six types of input: Contours, Points, Sinks, Streams, Lakes, and Boundaries. As many inputs can be used as desired, within reason. The order in which the inputs are entered does not have any bearing on the outcome. <Path> indicates a path to a dataset, <Item> indicates a field name, and <#> indicates a value to be entered.

- Contours—Contour line dataset with item containing height values.
- Points—Point dataset with item containing height values.
- Sinks—Point dataset containing sink locations. If the dataset has elevation values for the sinks, specify that field name as the <Item>. If only the locations of the sinks are to be used, use “NONE” for <Item>.
- Streams—Stream line dataset. Height values are not necessary.
- Lakes—Lake polygon dataset. Height values are not necessary.
- Boundary—Boundary polygon dataset. Height values are not necessary.
- Enforce—Controls whether drainage enforcement is applied.
- Datatype—Primary type of input data.
- Iterations—The maximum number of iterations the algorithm performs.
- Roughness Penalty—The measure of surface roughness.
- Discretization Error Factor—The amount to adjust the data smoothing of the input data into a raster.
- Vertical Standard Error—The amount of random error in the z-values of the input data.
- Tolerances—The first reflects the accuracy of elevation data in relation to surface drainage, and the other prevents drainage clearance through unrealistically high barriers.
- Z-Limits—Lower and upper height limits.
- Extent—Minimum x-, minimum y-, maximum x-, and maximum y-coordinate limits.
- Cell Size—The resolution of the final output raster.
- Margin—Distance in cells to interpolate beyond the specified output extent and boundary.
- Output Stream Features—Only use if Output stream polyline features is set in the Topo To Raster By File dialog box.
- Output Sink Features—Only use if Output remaining sink point features is set in the Topo To Raster By File dialog box.
- Output Diagnostics File—The location and name of the diagnostics file.
- An example parameter file is:


```
Contour D:\data\contours2\arc HEIGHT
Point D:\data\points2\point SPOTS
Sink D:\data\sinks_200.shp
Stream D:\data\streams\arc
Lake D:\data\lakes\polygon
Boundary D:\data\clipcov\polygon
ENFORCE ON
DATATYPE CONTOUR
ITERATIONS 40
ROUGHNESS_PENALTY 0.0000000000
DISCRETE_ERROR_FACTOR 1.0000000000
VERTICAL_STANDARD_ERROR 0.0000000000
TOLERANCES 2.5000000000 100.0000000000
ZLIMITS -2000.0000000000 13000.0000000000
EXTENT -810480.625000000000 8321785.000000000000 810480.625000000000
10140379.000000000000
CELL_SIZE 1800.000000000000
MARGIN 20
OUT_STREAM
OUT_SINK
OUT_DIAGNOSTICS D:\data\ttr_diag.txt
```

Trend: interpolates a surface from points using a trend technique.

`Trend <in_point_features> <z_field> <out_raster> {cell_size} {order} {LINEAR | LOGISTIC}`

- As the order of the polynomial is increased, the surface being fitted becomes progressively more complex. A higher-order polynomial will not always generate the most accurate surface; it is dependent on the data.
- The optional RMS file output contains information on the RMS (root mean square) error of the interpolation. This information can be used to determine the best value to specify for the polynomial order, by changing the order value until you get the lowest RMS error.
- For the LOGISTIC option of Type of Regression, the z-value field of input point features should have codes of 0 and 1.



Local toolset

Contains tools to compute an output raster dataset where the output value at each location is a function of the value associated with that location on one or more raster datasets.

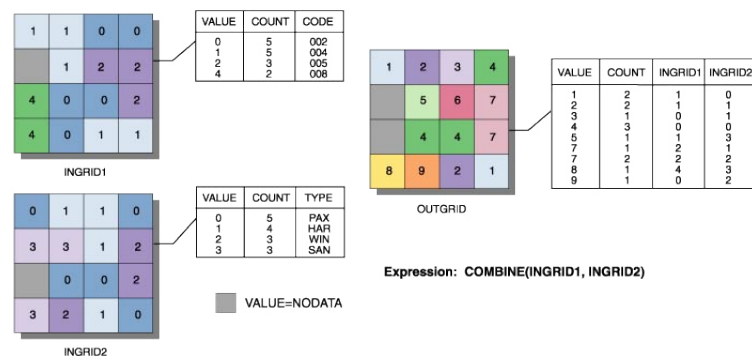
Cell Statistics: calculates a per cell statistic from multiple raster datasets.

`CellStatistics <in_rasters_or_constants;in_rasters_or_constants...> <out_raster>`
 {MEAN | MAJORITY | MAXIMUM | MEDIAN | MINIMUM | MINORITY | RANGE | STD | SUM | VARIETY}

- The order of input rasters is irrelevant.

Combine: combines multiple raster datasets on a cell-by-cell basis.

`Combine <in_rasters;in_rasters...> <out_raster>`



- Combine is similar to Combinatorial Or. They both assign a new number to each unique combination of input values.
- Combine works on integer values and their associated attribute tables. If the values on the input are floating point, they will be automatically truncated, tested for uniqueness with the other input, and sent to the output attribute table.
- No more than 20 rasters can be used as input to Combine.

Equal To Frequency: evaluates the number of times the input raster dataset values are equal to a specified value on a cell-by-cell basis.

`EqualToFrequency <in_value_raster> <in_rasters;in_rasters...> <out_raster>`



- An arbitrary number of rasters can be specified in the input rasters list.

Greater Than Frequency: evaluates the number of times the input raster dataset values are greater than a specified value on a cell-by-cell basis.

`GreaterThanFrequency <in_value_raster> <in_rasters;in_rasters...> <out_raster>`



- An arbitrary number of rasters can be specified in the input rasters list.

Highest Position: determines the position of a raster dataset with the maximum value in a set of raster datasets.

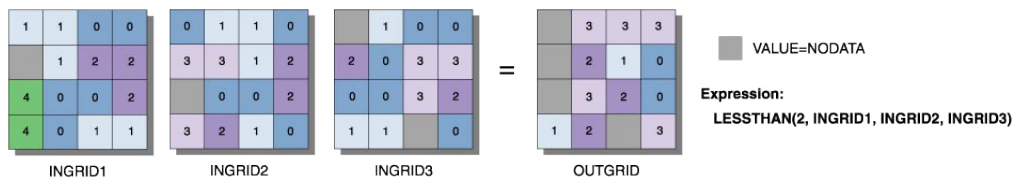
`HighestPosition <in_rasters_or_constants;in_rasters_or_constants...> <out_raster>`



- An arbitrary number of rasters can be specified in the input rasters list.
- The order of inputs is relevant for the Highest Position tool.
- If two or more input rasters contain the maximum value for a particular cell location, the position of the first one is returned on the output raster.

Less Than Frequency: evaluates the number of times the input raster dataset values are less than a specified value on a cell-by-cell basis.

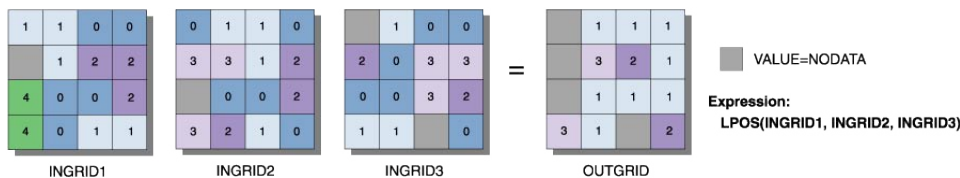
`LessThanFrequency <in_value_raster> <in_rasters;in_rasters...> <out_raster>`



- An arbitrary number of rasters can be specified in the input rasters list.

Lowest Position: determines the position for each cell of the input raster dataset with the minimum value in the argument list.

`LowestPosition <in_rasters_or_constants;in_rasters_or_constants...> <out_raster>`



- An arbitrary number of rasters can be specified in the input rasters list.
- The order of inputs is relevant for the Lowest Position tool.
- If two or more input rasters contain the minimum value for a particular cell location, the position of the first one encountered is returned on the output raster.

Popularity: determines the value that is at a specified level of popularity on a cell-by-cell basis.

Popularity <in_popularity_raster_or_constant> <in_rasters;in_rasters...> <out_raster>



- An arbitrary number of rasters can be specified in the input rasters list.
- The order of input rasters is irrelevant.

Rank: returns the value of a set of raster datasets based on a rank level specified by another raster dataset on a cell-by-cell basis.

Rank <in_rank_raster_or_constant> <in_rasters;in_rasters...> <out_raster>



- An arbitrary number of rasters can be specified in the input rasters list.
- The order of input rasters is irrelevant.
- If the input values are all the same for any cell location, no matter what the specified popularity is, the output value will be the same as the input for that cell location.



Map Algebra toolset

Contains the tool to create expressions using any of the Map Algebra statements. Map Algebra is the analysis language from Spatial Analyst.

Multi Output Map Algebra: executes GRID's Map Algebra statements.

MultiOutputMapAlgebra <expression_string>

- The Map Algebra expression requires that the output dataset name be identified. For example, the expression must be entered as "out_slope = slope (D:\data\surf_1, percentrise, 2)".
- Only ArcInfo GRIDs are supported for use with Multi Output Map Algebra.
- Inputs must be in the same spatial reference to be used with Multi Output Map Algebra.
- The results from Multi Output Map Algebra are not added to the table of contents for the active ArcMap session.

Single Output Map Algebra: executes GRID's Map Algebra statement to produce a raster dataset.

SingleOutputMapAlgebra <expression_string> <out_raster> {in_data;in_data...}

- The Map Algebra expression does not support the out_dataset = function (parameters) syntax of traditional Map Algebra. Simply specify the function and its parameters. For example, type "slope (D:\data\surf_1, percentrise, 2)" instead of "out_slope = slope (D:\data\surf_1, percentrise, 2)".
- If the dataset is identified in the input raster or feature data list, it is not necessary to specify the path in the Map Algebra expression.
- Inputs with different spatial references can be used with Single Output Map Algebra. The datasets will be projected on the fly to complete the analysis.

- The results from Single Output Map Algebra are added to the table of contents for the active ArcMap session.



Math toolset

Contains tools to implement math functions, which apply a specified mathematical operation or function to each cell location on an input raster or series of raster datasets.

Abs: calculates the absolute value of the input raster dataset on a cell-by-cell basis.

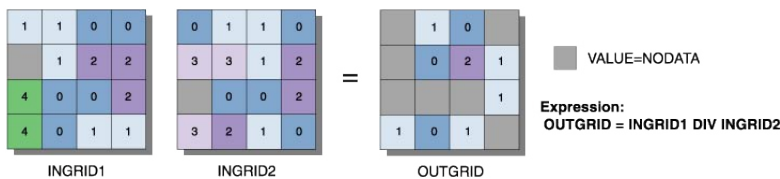
Abs <in_raster_or_constant> <out_raster>



- Input values can be positive or negative and can be either integer or floating point.

Divide: divides the values of two input raster datasets on a cell-by-cell basis.

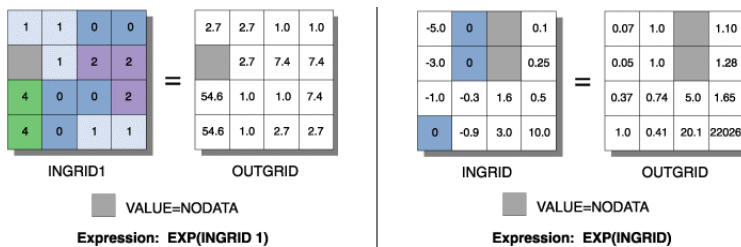
Divide <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- The order of input is relevant for Divide.
- When a number is divided by 0, the output result is NoData.
- If both inputs are integers, then Divide performs an integer division and the output result is an integer. For example, if 3 is to be divided by 2, the output is 1.
- If either input is of floating-point type, then Divide performs a floating-point division and the result is a floating-point value. For example, if 3 is divided by 2.0, the output is 1.5.

Exp: calculates the base e exponential of cells in a raster dataset.

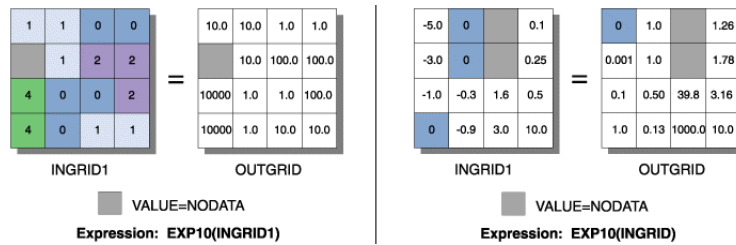
Exp <in_raster_or_constant> <out_raster>



- The base e exponential is the most commonly used exponential function.

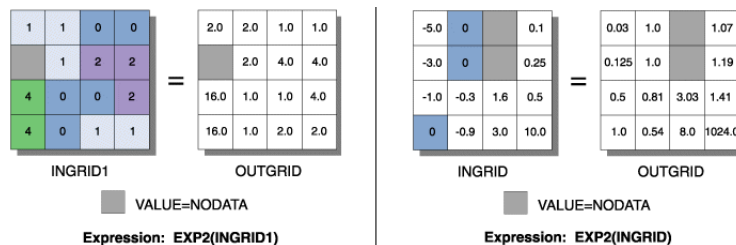
Exp10: calculates the base 10 exponential of cells in a raster dataset.

Exp10 <in_raster_or_constant> <out_raster>



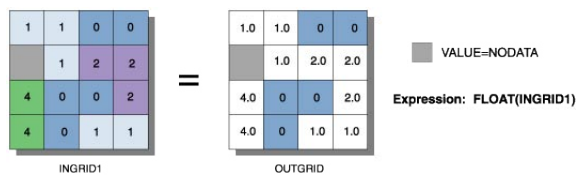
Exp2: calculates the base 2 exponential of cells in a raster dataset.

Exp2 <in_raster_or_constant> <out_raster>



Float: converts each cell value in a raster dataset to floating-point values.

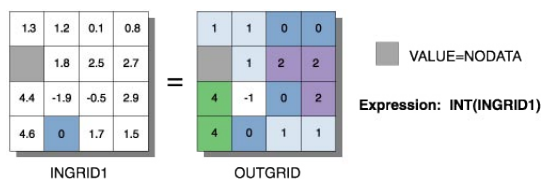
Float <in_raster_or_constant> <out_raster>



- Input values are integers and can be positive or negative.

Int: converts each cell value in a raster dataset to an integer by truncation.

Int <in_raster_or_constant> <out_raster>

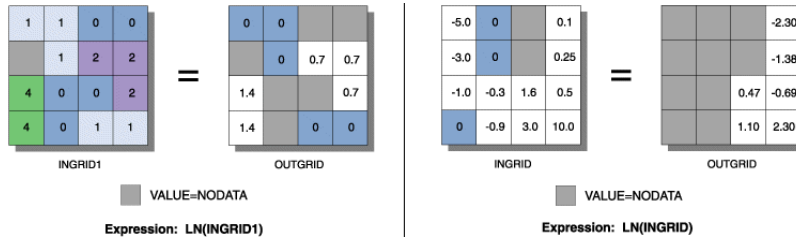


- If rounding is preferred rather than truncating, add a 0.5 input raster prior to performing **Int**.
- The difference between the Round Down and **Int** functions is that **Int** always truncates a number:
 - int** on 1.5 becomes 1
 - int** on -1.5 becomes -1
 - while for the same two values, Round Down returns
 - round down on 1.5 becomes 1.0
 - round down on -1.5 becomes -2.0

A second difference between the two functions is that Round Down outputs floating-point values, and **Int** outputs integer values.

Ln: calculates the natural logarithm (base e) of cells in a raster dataset.

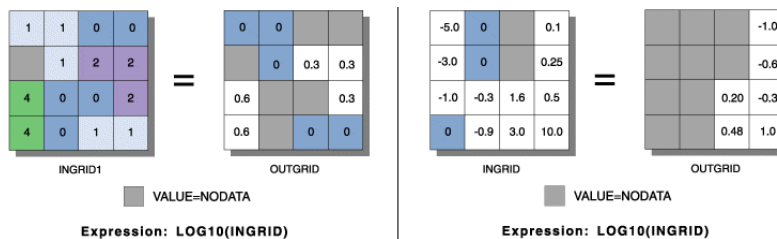
Ln <in_raster_or_constant> <out_raster>



- The natural logarithm is the most commonly used logarithmic function.
- The input value of a logarithmic function cannot be 0 or a negative number. If it is, the output will be NoData.

Log10: calculates the base 10 logarithm of cells in a raster dataset.

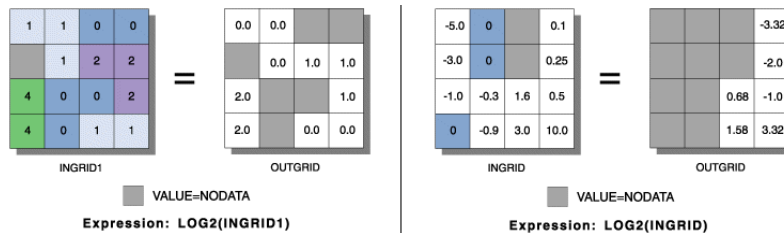
Log10 <in_raster_or_constant> <out_raster>



- The input value of a logarithmic function cannot be 0 or a negative number. If it is, the output will be NoData.

Log2: calculates the base 2 logarithm of cells in a raster dataset.

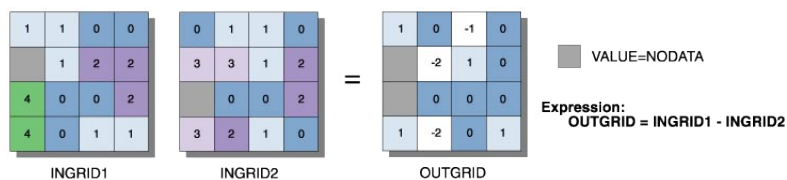
Log2 <in_raster_or_constant> <out_raster>



- The input value of a logarithmic function cannot be 0 or a negative number. If it is, the output will be NoData.

Minus: subtracts the values of the second input from the values of the first input on a cell-by-cell basis.

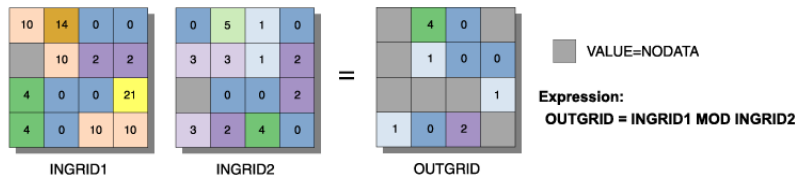
Minus <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- The order of input is relevant in the subtraction expression.

Mod: divides the values of the first input by the values of the second input and returns the remainder.

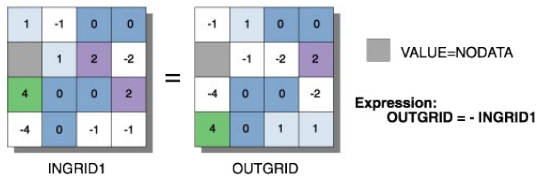
Mod <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- The order of input is relevant in the modulus expression.
- Any value modulated (divided) by 0 is assigned NoData on the output. Therefore, any location on the second input that is either 0 or NoData will return NoData for that location on the output.
- Mod assumes both its inputs are integers. If any of the inputs are not integers, those inputs will be converted to integers through truncation. Output values are always integers.

Negate: changes the sign of the cell values of the input raster dataset (multiplies by -1).

Negate <in_raster_or_constant> <out_raster>



Plus: adds the values of two raster datasets on a cell-by-cell basis.

Plus <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- The order of input is irrelevant in the addition expression.

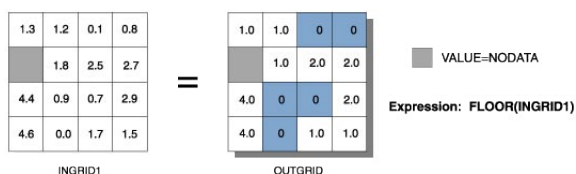
Power: calculates the nth power of the input raster or number on a cell-by-cell basis.

Power <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



Round Down: returns the next lower whole number value for each cell in a raster dataset.

RoundDown <in_raster_or_constant> <out_raster>



- In Round Down, if a number has any values to the right of the decimal point, the output will be assigned the next lowest whole value:

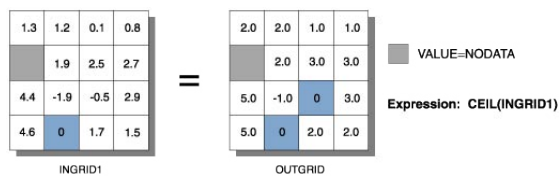
Input	Output
5.3	5.0
4.9	4.0
3.0	3.0
6.5	6.0
-0.2	-1.0
-2.8	-3.0

- The difference between Round Down and Int is that Int always truncates a number:
int on 1.5 becomes 1
int on -1.5 becomes -1
while Round Down returns the next lower whole number:
Round Down on 1.5 becomes 1.0
Round Down on -1.5 becomes -2.0

A second difference is that Round Down outputs floating-point values while Int outputs integer values.

Round Up: returns the next highest whole number value that is greater than or equal to the input value for each cell in a raster dataset.

RoundUp <in_raster_or_constant> <out_raster>



- In Round Up, if a number has any values to the right of the decimal point, the output will be assigned the next highest whole value:

Input	Output
5.3	6.0
4.9	5.0
3.0	3.0
6.5	7.0
-0.2	0.0
-2.8	-2.0

Square: calculates the square of cell values in a raster dataset.

Square <in_raster_or_constant> <out_raster>



Square Root: calculates the square root of the input grid or number for each cell.

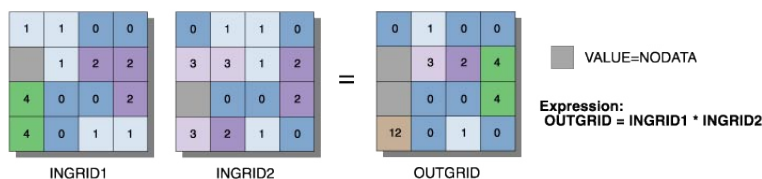
SquareRoot <in_raster_or_constant> <out_raster>



- Input values must be greater than or equal to 0. If they are not, the output will be NoData.

Times: multiplies the values of two raster datasets on a cell-by-cell basis.

`Times <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- The order of input is irrelevant in the multiplication expression.



Bitwise (Math) toolset

Contains tools to implement bitwise operators, which treat the operands as bits (binary representations) and calculate the output by applying the logical operation (e.g., 3 BITWISEAND 5 = 0011 && 0101 = 0001 = 1).

Bitwise And: performs the Bitwise AND operation on the binary values of two inputs on a cell-by-cell basis (e.g., 0011 && 0101 = 0001).

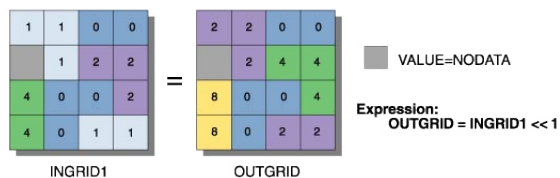
`BitwiseAnd <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- The bitwise methods work on 32-bit integers.
- If floating-point values are input, they are converted to integer values through truncation before the bitwise method is performed. The output values are always integers.
- Binary values are stored in two's complement.
- The leftmost bit position is reserved for the sign of the value (positive or negative). If the integer is positive, the bit is 0; if negative, the bit is 1.
- The Bitwise And tool treats the sign bit as it would any other bit. If one or both inputs for a cell location are positive, the output is positive; if both inputs are negative, the output is negative.

Bitwise Left Shift: shifts the bits to the left using the number specified (e.g., 1<<2 = 0001 << 2 = 0100 = 4).

`BitwiseLeftShift <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- The bitwise operators work on 32-bit integers.
- If floating-point values are input, they are converted to integer values through truncation before the bitwise operation is performed. The output values are always integer.
- Binary values are stored in two's complement.
- The leftmost bit position is reserved for the sign of the value (positive or negative). If the integer is positive, the bit position is 0; if negative, the bit position is 1.

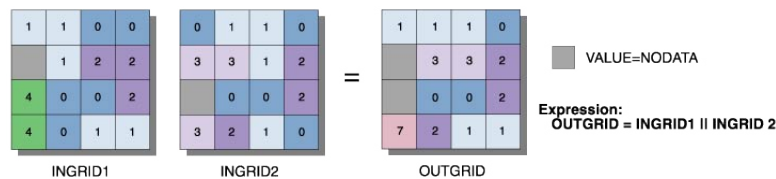
Bitwise Not: performs the bitwise COMPLEMENT operation on the binary values of two inputs on a cell-by-cell basis (flips the bits; e.g., 5 = 0101 ~ 0101 = 10).

`BitwiseNot <in_raster_or_constant> <out_raster>`

- The bitwise operators work on 32-bit integers.
- If floating-point values are input, they are converted to integer values through truncation before the bitwise operation is performed. The output values are always integer.
- Binary values are stored in two's complement.
- The leftmost bit position is reserved for the sign of the value (positive or negative). If the integer is positive, the bit position is 0; if negative, the bit position is 1.
- The Bitwise Not tool treats the sign bit as it would any other bit. If one or both inputs for a cell location are negative, the output is negative; if both inputs are positive, the output is positive.

Bitwise Or: performs a bitwise OR operation on the binary values of two inputs on a cell-by-cell basis (e.g., 0101 || 1100 = 1101).

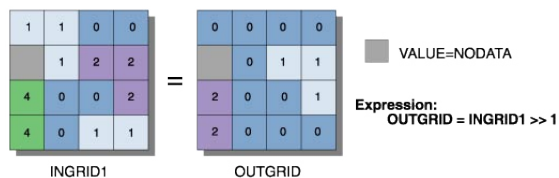
`BitwiseOr <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- The bitwise operators work on 32-bit integers.
- If floating-point values are input, they are converted to integer values through truncation before the bitwise operation is performed. The output values are always integer.
- Binary values are stored in two's complement.
- The leftmost bit position is reserved for the sign of the value (positive or negative). If the integer is positive, the bit position is 0; if negative, the bit position is 1.

Bitwise Right Shift: shifts the bits to the right using the number specified (e.g., 6 >> 1 = 0110 >> 1 = 0011 = 3).

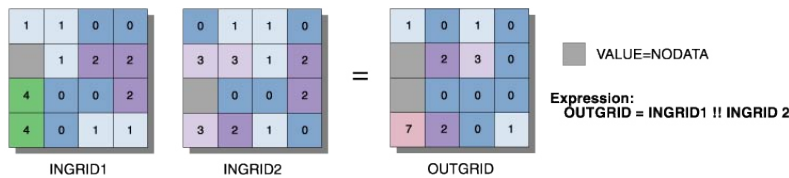
`BitwiseRightShift <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- The bitwise operators work on 32-bit integers.
- If floating-point values are input, they are converted to integer values through truncation before the bitwise operation is performed. The output values are always integer.
- Binary values are stored in two's complement.
- The leftmost bit position is reserved for the sign of the value (positive or negative). If the integer is positive, the bit position is 0; if negative, the bit position is 1.
- The Bitwise Right Shift operation does no wrapping of bits. The rightmost bit is dropped off.

Bitwise XOR: performs a bitwise exclusive OR operation on the binary values of two inputs on a cell-by-cell basis (e.g., 0101 !! 1100 = 1001).

`BitwiseXor <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- The bitwise operators work on 32-bit integers.
- If floating-point values are input, they are converted to integer values through truncation before the bitwise operation is performed. The output values are always integer.
- Binary values are stored in two's complement.
- The leftmost bit position is reserved for the sign of the value (positive or negative). If the integer is positive, the bit position is 0; if negative, the bit position is 1.
- The Bitwise XOR treats the sign bit as it would any other bit. If one or both inputs for a cell location are negative, the output is negative; if both inputs are positive, the output is positive.

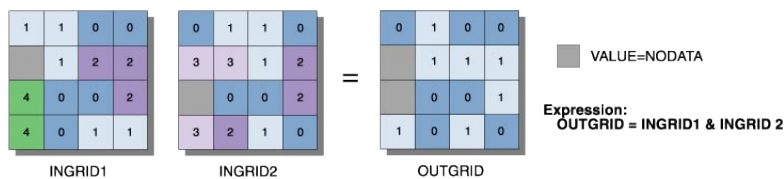


Logical (Math) toolset

Contains tools to evaluate the values of an input raster or rasters relative to a conditional statement, the values in another raster, a constant value, or a specific value. Also contains tools that can produce an output that tracks the unique combinations of the input values between two rasters or constants.

Boolean And: performs the Boolean AND operator on the cell values of two input raster datasets.

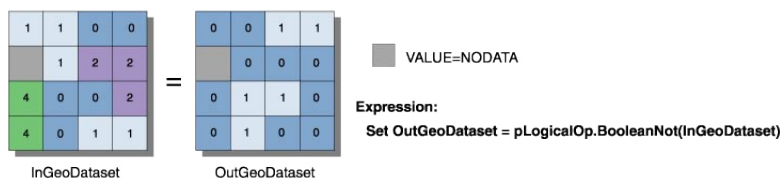
`BooleanAnd <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- If the input values are floating point, they are converted to integer values by truncation before the Boolean operation is performed. The output values are always integers.
- Boolean And interprets the input as Boolean values, where nonzero values are considered True and zero is considered False. The two input rasters are tested on a cell-by-cell basis. If both values are True, the output is 1. If one or both values are False, the output is 0. If one or both values are NoData, the output value is NoData.

Boolean Not: performs the Boolean COMPLEMENT operator on the cell values of two input raster datasets.

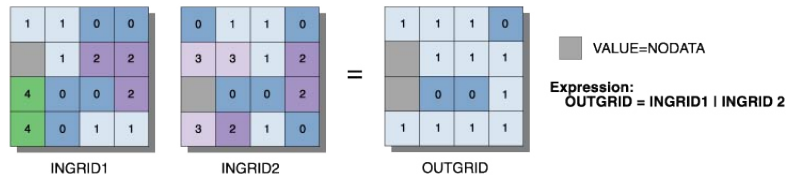
`BooleanNot <in_raster_or_constant> <out_raster>`



- If the input values are floating point, they are converted to integers by truncation before the Boolean Not is performed. The output values are always integers.
- Boolean Not interprets the input as Boolean values, where nonzero values are considered True and zero is considered False. The input raster is tested on a cell-by-cell basis. If the value is True, the output is 0 (the complement of True). If the value is False, the output is 1. If the value is NoData, the output value is NoData.

Boolean Or: performs the Boolean Or operator on the cell values of two input raster datasets.

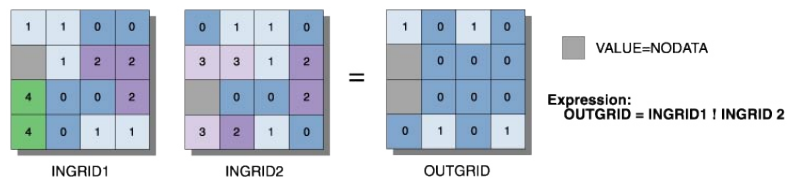
`BooleanOr <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- If the input values are floating point, they are converted to integer values by truncation before the Boolean method is performed. Output values are always integers.
- Boolean Or interprets the input as Boolean values, where nonzero values are considered True and zero is considered False. The two input rasters are tested on a cell-by-cell basis. If one or both values are True, the output is 1. If both values are False, the output is 0. If one or both values are NoData, the output value is NoData.

Boolean XOr: performs the Boolean exclusive Or operator on the cell values of two input raster datasets.

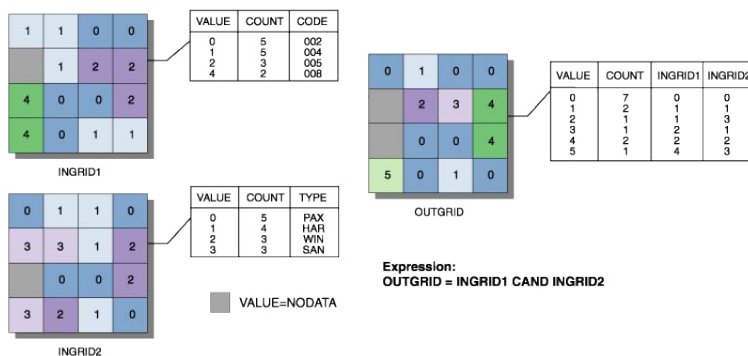
`BooleanXOr <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- If the input values are floating point, they are converted to integer values by truncation before the Boolean method is performed. The output values are always integers.
- Boolean XOr interprets the input as Boolean values, where nonzero values are considered True and zero is considered False. The two input rasters are tested on a cell-by-cell basis. If one value is True and one value is False, the output is 1. If both values are True or both are False, the output is 0. If one or both values are NoData, the output value is NoData.

Combinatorial And: performs a combinatorial AND operation on two input raster datasets.

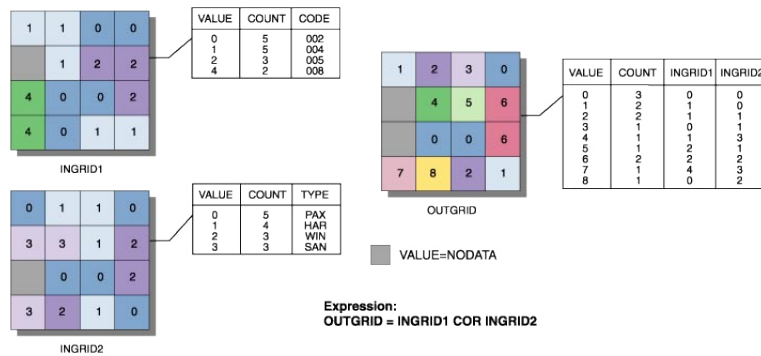
`CombinatorialAnd <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- Both inputs must contain positive integer values.

Combinatorial Or: performs a combinatorial OR operation on the cell values of two input raster datasets.

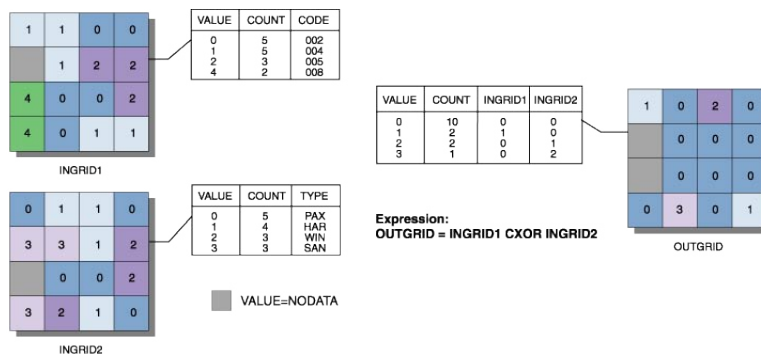
`CombinatorialOr <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- Both inputs must contain positive integer values.

Combinatorial XOR: performs a combinatorial exclusive OR operation on the cell values of two input raster datasets.

CombinatorialXOR <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- Both inputs must contain positive integer values.

Equal To: returns 1 for cells where the first raster equals the second raster and 0 if it does not.

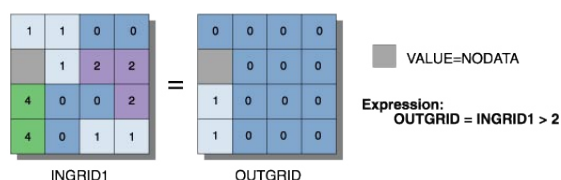
EqualTo <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- Equal To evaluates the first input value in relation to the second input value on a cell-by-cell basis within the Analysis window. In the relational evaluation, if the condition is True (the first and second input values are the same), the output is 1; if it is False, the output is 0. When one or both input values are NoData, the output is NoData.

Greater Than: returns 1 for cells where the first raster dataset is greater than the second raster dataset and returns 0 where it is not.

GreaterThan <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>

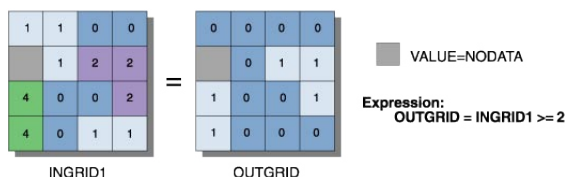


- The order of input is relevant in the greater than expression.

- **Greater Than** evaluates the first input value in relation to the second input value on a cell-by-cell basis within the Analysis window. In the relational evaluation, if the condition is True (the first input value is greater than the second input value), the output is 1; if it is False, the output is 0. When one or both input values are NoData, the output is NoData.

Greater Than Equal: returns 1 for cells where the first raster dataset is greater than or equal to the second raster dataset and returns 0 where it is not.

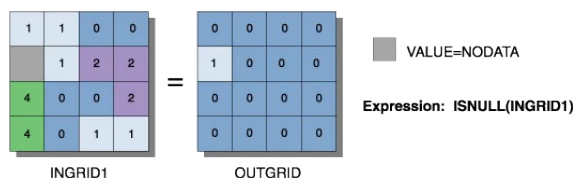
Greater Than Equal <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- The order of input is relevant in the greater than equal expression.
- **Greater Than Equal** evaluates the first input value in relation to the second input value on a cell-by-cell basis within the Analysis window. In the relational evaluation, if the condition is True (the first input value is greater than or equal to the second input value), the output is 1; if it is False, the output is 0. When one or both input values are NoData, the output is NoData.

Is Null: returns 1 for cells in the input raster dataset that have a value of NoData and returns 0 where they do not.

Is Null <in_raster> <out_raster>



- The output value type is always integer. The values are either 1 or 0. Cells in the input that have a value are given 0 on the output. Cells that are NoData in the input are given a value of 1 on the output.

Less Than: returns 1 for cells where the first raster dataset is less than the second raster dataset and returns 0 where it is not.

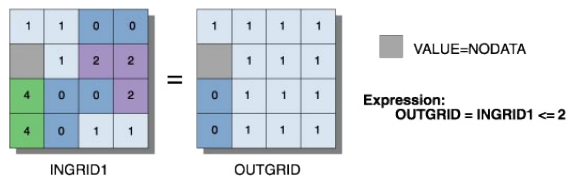
Less Than <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- The order of input is relevant in the less than expression.
- **Less Than** evaluates the first input value in relation to the second input value on a cell-by-cell basis within the Analysis window. In the relational evaluation, if the condition is True (the first input value is less than the second input value), the output is 1; if it is False, the output is 0. When one or both input values are NoData, the output is NoData.

Less Than Equal: returns 1 for cells where the first raster dataset is less than or equal to the second raster dataset and returns 0 where it is not.

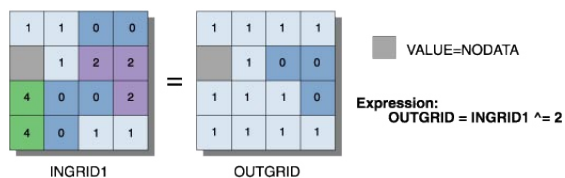
Less Than Equal <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- The order of input is relevant in the less than or equal to expression.
- Less Than Equal evaluates the first input value in relation to the second input value on a cell-by-cell basis within the Analysis window. In the relational evaluation, if the condition is True (the first input value is less than or equal to the second input value), the output is 1; if it is False, the output is 0. When one or both input values are NoData, the output is NoData.

Not Equal: returns 1 for cells where the first raster dataset is not equal to the second raster dataset and returns 0 where it is not.

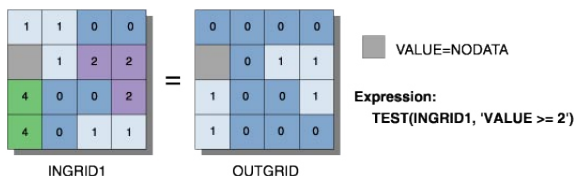
NotEqual <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- Two inputs are necessary for Not Equal.
- The order of input is irrelevant for Not Equal.
- Not Equal evaluates the first input value in relation to the second input value on a cell-by-cell basis within the Analysis window. In the relational evaluation, if the condition is True (the first input is not equal to the second input), the output is 1; if it is False, the output is 0. When one or both input values are NoData, the relational expression outputs NoData.

Test: returns 1 for cells that evaluate to true based on a logical expression and returns 0 for cells that evaluate to false.

Test <in_raster> <where_clause> <out_raster>



- The output is either 1 (if the test evaluates to True) or 0 (if the test evaluates to False).
- The test is specified by a SQL expression.

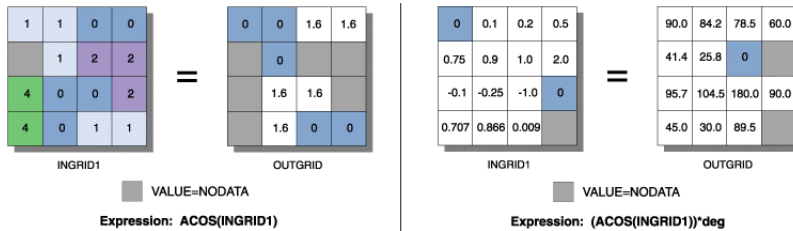


Trigonometric (Math) toolset

Contains tools for the trigonometric functions that are applied on a per-cell basis to an input raster dataset.

ACos: calculates the inverse cosine of the input raster dataset or number on a cell-by-cell basis.

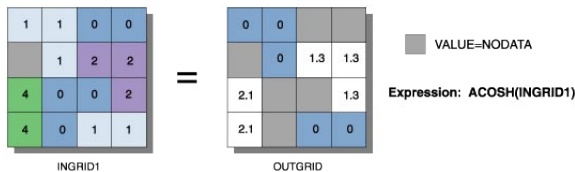
ACos <in_raster_or_constant> <out_raster>



- The input values to the ACos function must be equal to or between -1 and 1. Any value outside this range will receive NoData on the output raster.
- The input values to the ACos function are interpreted as unitless.
- The output values from ACos are in radians. If degrees are desired, the resulting raster must be multiplied by the radians-to-degrees conversion factor of $180/\pi$, or approximately 57.296.

ACosH: calculates the inverse hyperbolic cosine of cells in an input raster dataset.

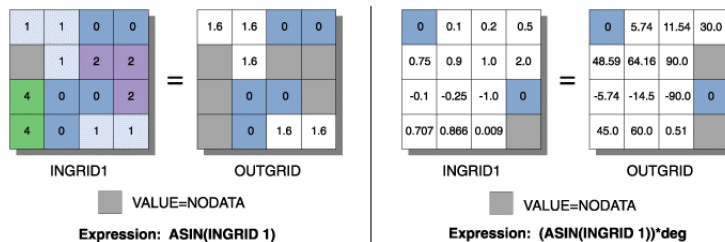
ACosH <in_raster_or_constant> <out_raster>



- The input values to the ACosH function must be greater than or equal to 1. Any value below 1 will receive NoData on the output.
- The input and output values in ACosH are interpreted as unitless.

ASin: calculates the inverse sine of cells in an input raster dataset.

ASin <in_raster_or_constant> <out_raster>



- The input values to ASin must be between -1 and 1. Any value outside this range will receive NoData on the output.
- The input values to ASin are interpreted as unitless.
- The output values from ASin are in radians. If degrees are desired, the resulting raster must be multiplied by the radians-to-degrees conversion factor of $180/\pi$, or approximately 57.296.

ASinH: calculates the inverse hyperbolic sine of cells in an input raster dataset.

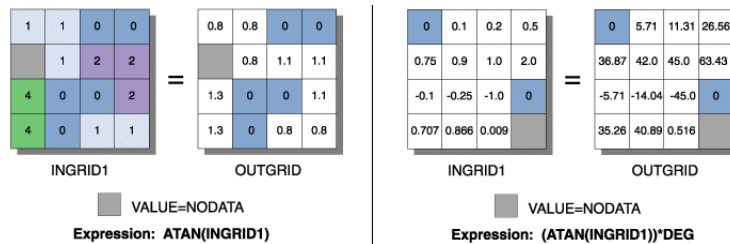
ASinH <in_raster_or_constant> <out_raster>



- Output values are always floating point, no matter what the input values are.
- There is no limit on the input values to ASinH.
- The input and output values in ASinH are interpreted as unitless.

ATan: calculates the inverse tangent of cells in an input raster dataset.

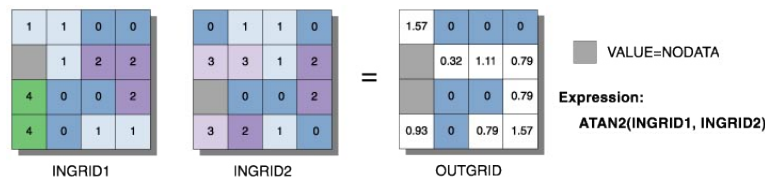
ATan <in_raster_or_constant> <out_raster>



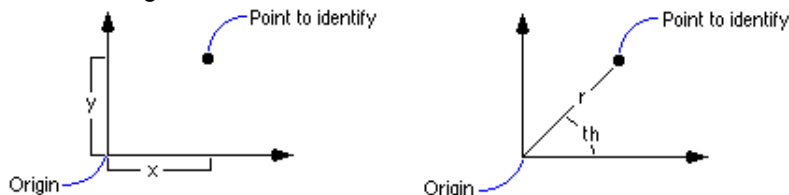
- The input values to the ATan function are interpreted as unitless.
- The output values from ATan are in radians. If degrees are desired, the resulting raster must be multiplied by the radians-to-degrees conversion factor of $180/\pi$, or approximately 57.296.

ATan2: calculates the inverse tangent (based on y/x) of cells in an input raster dataset.

ATan2 <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



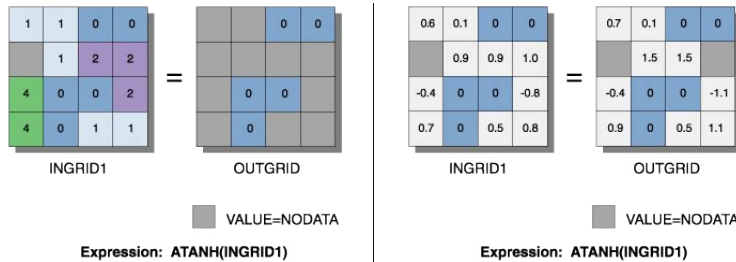
- The values of the first specified input are used as the numerator in the calculation of the tangent angle ($\tan@ = y/x$). The values of the second specified input are used as the denominator in the calculation of the angle.
- The output values from the ATan2 function are between $-\pi$ and π . The arc tangent two operation represents all quadrants in a Cartesian matrix (based on sign).
- ATan2 converts rectangular coordinates (x,y) to polar (r,th), where r is the distance from the origin and th is the angle from the x-axis.



- The input values to ATan2 are interpreted as unitless.
- The output values from ATan2 are in radians. If degrees are desired, the resulting raster must be multiplied by the radians-to-degrees conversion factor of $180/\pi$, or approximately 57.296.

ATanH: calculates the inverse hyperbolic tangent of cells in an input raster dataset.

ATanH <in_raster_or_constant> <out_raster>



- The input and output values in ATanH are interpreted as unitless.
- The input values to ATanH must be between -1 and 1. Any input value outside this range will result in NoData on the output raster.

Cos: calculates the cosine of cells in an input raster dataset.

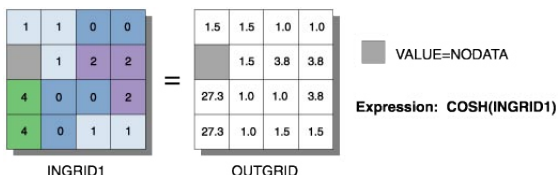
Cos <in_raster_or_constant> <out_raster>



- The cosine of a value is between -1 and 1.
- The input values to Cos are interpreted as radians. If the desired input is in degrees, the values must be divided by the radians-to-degrees conversion factor of $180/\pi$, or approximately 57.296.

CosH: calculates the hyperbolic cosine of cells in an input raster dataset.

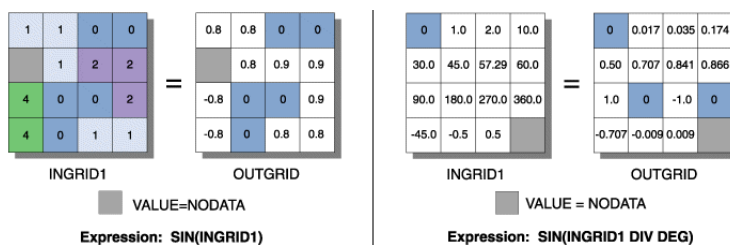
CosH <in_raster_or_constant> <out_raster>



- The input and output values in CosH are interpreted as unitless.

Sin: calculates the sine of cells in an input raster dataset.

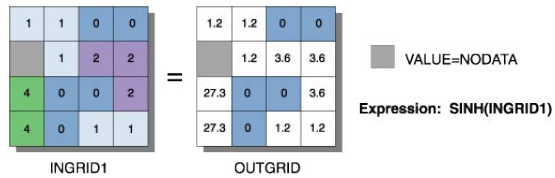
Sin <in_raster_or_constant> <out_raster>



- The sine of a value is between -1 and 1.
- The input values to Sin are interpreted as radians. If the desired input is in degrees, the values must be divided by the radians-to-degrees conversion factor of $180/\pi$, or approximately 57.296.

SinH: calculates the hyperbolic sine of cells in an input raster dataset.

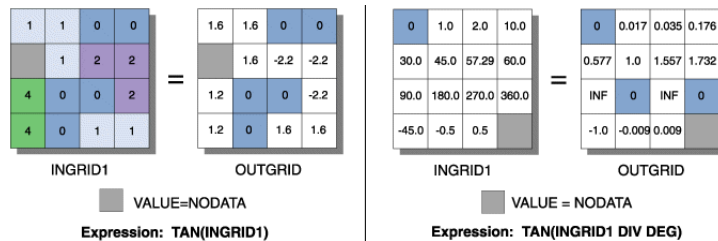
`SinH <in_raster_or_constant> <out_raster>`



- The input and output values in the SinH function are interpreted as unitless.

Tan: calculates the tangent of cells in an input raster dataset.

`Tan <in_raster_or_constant> <out_raster>`



- The input values to Tan are interpreted as radians. If the desired input is in degrees, the values must be divided by the radians-to-degrees conversion factor of $180/\pi$, or approximately 57.296.

TanH: calculates the hyperbolic tangent of cells in an input raster dataset.

`TanH <in_raster_or_constant> <out_raster>`

- The input and output values of TanH are interpreted as unitless.



Multivariate toolset

Contains tools to allow the statistical analysis of a series of raster datasets (independent variables) that, based on the values within the rasters, produces a predictable result for some phenomena (the dependent variable).

Band Collection Statistics: calculates the statistics for a set of raster bands.

`BandCollectionStats <in_raster_bands; in_raster_bands...> <out_stat_file>`
`{BRIEF | DETAILED}`

- The output is saved to an ASCII file named the output statistics file. Any extension can be used, but .txt is preferred.
- By default the minimum, maximum, mean, and standard deviation of the input raster bands are calculated. If the extents of the raster bands are not the same, the statistics will be calculated on the common spatial extent of all the input raster bands. The cell size will be that of the maximum of the input rasters, by default. Otherwise, it will depend on the Raster Analysis Setting in the environment.

Class Probability: creates probability layers for each class in a signature file.

`ClassProbability <in_raster_bands; in_raster_bands...> <in_signature_file>`
`<out_multiband_raster> {maximum_output_value} {EQUAL | SAMPLE | FILE} {in_a_priori_file}`

- Any signature file created by one of the Create Signature, Edit Signature, or Iso Cluster tools is a valid entry for input signature file. These will have a .gsg extension.
- The value entered for maximum output value sets the upper range of the values in the output probability layers. The default value of 100 creates a stack with each layer containing integer values ranging from 0 to 100.

- The Input a priori probability file must be an ASCII file consisting of two columns. The values in the left column represent class IDs. The values in the right column represent the a priori probabilities for the respective classes. Valid values for class a priori probabilities must be greater than or equal to 0. If 0 is specified as a probability, the class will not appear on the output raster. The sum of the specified a priori probabilities must be smaller than or equal to 1. The format of the file is as follows:

```
1 .3
2 .1
4 .0
5 .15
7 .05
8 .2
```

The classes omitted in the file will receive the average a priori probability of the remaining portion of the value of 1.

Create Signatures: creates an ASCII signature file of classes defined by input sample data and a set of raster bands.

```
CreateSignatures <in_raster_bands;in_raster_bands...> <in_sample_data>
<out_signature_file> {COVARIANCE | MEAN_ONLY} {sample_field}
```

- The minimum valid number of class samples in the sample data is 2. There is no maximum number of classes.
- The {COVARIANCE} option must be used if the signature file is to be used in further multivariate analysis functions then use covariance matrices, such as Maximum Likelihood Classification and Class Probability. This is the default.

Dendrogram: constructs a tree diagram showing attribute distances between sequentially merged classes in a signature file.

```
Dendrogram <in_signature_file> <out_dendrogram_file> {VARIANCE | MEAN_ONLY} {line_width}
```

- The Input signature file has to be a signature file in the prescribed ASCII format. A signature file can be created with the Iso Cluster or Create Signatures tool. The file must have a minimum of two classes.
- The output of Dendrogram, the output dendrogram file, is an ASCII file. The file contains a table of distances between pairs of sequentially merged classes and a graphical representation showing the relationships among classes and the hierarchy of the merging. By analyzing the graph and the associated table, a user can decide the potential merging of classes.
- The proximity of a pair of classes within a signature file is measured by the attribute distance.

Edit Signatures: edits and updates a signature file by merging, renumbering, and deleting class signatures.

```
EditSignatures <in_raster_bands;in_raster_bands...> <in_signature_file>
<in_signature_remap_file> <out_signature_file> {sample_interval}
```

- Edit Signatures allows the modification of an existing signature file by all or any of the following operations:
 - Merging signatures of a set of classes
 - Renumbering a signature class ID
 - Deleting unwanted signatures
- The input signature file has to be an ASCII signature file. The file can be output of any Multivariate function that produces the file containing the required statistical information (for example, Iso Cluster and Create Signatures). The file must have a minimum of two classes. Such a file can be recognized by its .gsg extension.

- The input signature remap file is an ASCII file consisting of two columns. In the first column, the original class IDs are listed in ascending order. The second column has the new class IDs for updating in the signature file. When a set of classes is to be merged, a new class ID must be put in the second column for each class ID of the set. Only classes that need to be edited have to be placed in the signature remap file. Any class not present in the remap file will remain unchanged. To delete a class signature, the value of -9,999 must be entered in the second column of the remap file. A class ID can also be renumbered to a value that does not exist in the input signature file. The following is an example of the input signature remap file:

```
2 : 3
4 : 11
5 : -9999
9 : 3
```

The example above will merge classes 2 and 9 with 3, class 4 with 11, and delete class 5.

Iso Cluster: uses isodata clustering to determine the characteristics of natural cell groupings in multidimension attribute space.

```
IsoCluster <in_raster_bands;in_raster_bands...> <out_signature_file> <number_classes>
{number_iterations} {min_class_size} {sample_interval}
```

- Iso Cluster performs clustering of the multivariate data combined in a list of input rasters. The resulting signature file may be used as the input for a classification function, such as Maximum Likelihood Classification, producing an unsupervised classification raster.
- To provide the sufficient statistics necessary to generate a signature file for a future classification, each cluster should contain enough cells to accurately represent the cluster. The value entered for the minimum class size should be approximately 10 times larger than the number of layers in the input raster bands.
- Generally, for more cells contained in the input rasters extent, larger values for minimum class size and sample interval should be specified. Values entered for the sample interval should be small enough that the smallest desirable categories existing in the input data will be appropriately sampled.
- The class ID values on the output signature file start from one and sequentially increase to the number of input classes. The assignment of the class numbers is arbitrary.
- Better results will be obtained if all input rasters have the same data ranges.

Maximum Likelihood Classification: performs a maximum likelihood classification on a set of raster dataset bands.

```
MLClassify <in_raster_bands;in_raster_bands...> <in_signature_file> <out_classified_
raster> {0.0 | 0.005 | 0.01 | 0.025 | 0.05 | 0.1 | 0.25 | 0.5 | 0.75 | 0.9 | 0.95 | 0.975 | 0.99
| 0.995} {EQUAL | SAMPLE | FILE} {in_a_priori_file} {out_confidence_raster}
```

- Any signature file created by the Create Signature, Edit Signature, or Iso Cluster tools is a valid entry for the input signature file. These will have a .gsg extension.
- By default, all cells in the output raster will be classified, with each class having equal probability weights attached to their signatures.
- The input a priori probability file must be an ASCII file consisting of two columns. The values in the left column represent class IDs. The values in the right column represent the a priori probabilities for the respective classes. Valid values for class a priori probabilities must be greater than or equal to 0. If 0 is specified as a probability, the class will not appear on the output raster. The sum of the specified a priori probabilities must be less than or equal to 1. The format of the file is as follows:

```
1 .3
2 .1
4 .0
5 .15
7 .05
8 .2
```


The classes omitted in the file will receive the average a priori probability of the remaining portion of the value of 1.

- A specified reject fraction, which lies between any two valid values, will be assigned to the next upper valid value. For example, 0.02 will become 0.025.

Principal Components: performs principal components analysis on a set of raster bands.

```
PrincipalComponents <in_raster_bands> <in_raster_bands...> <out_multiband_raster>
{number_components} {out_data_file}
```

- The value specified for the number of principal components determines the number of principal component layers in the output multiband raster. The number must not be larger than the total number of raster bands in the input.
- With the output data file name specified, the correlation and covariance matrices, as well as the eigenvalues and eigenvectors, will be stored in an ASCII file.



Neighborhood toolset

Contains tools to calculate a statistic or value based on the values at each processing cell and the values of the cells within an identified neighborhood.

Block Statistics: calculates statistics for a nonoverlapping neighborhood.

```
BlockStatistics <in_raster> <out_raster> {neighborhood} {MEAN | MAJORITY | MAXIMUM | MEDIAN
| MINIMUM | MINORITY | RANGE | STD | SUM | VARIETY} {DATA | NODATA}
```

- For statistics type Majority, cells where there is no single majority value (that is, two or more values within a block are tied as having the most number of cells with the value) will be assigned NoData. For statistics type Minority, cells where there is no single minority value will be similarly assigned NoData.

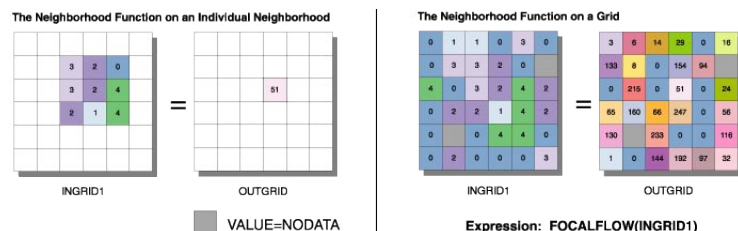
Filter: performs a preset focal filter on a raster.

```
Filter <in_raster> <out_raster> {LOW | HIGH} {DATA | NODATA}
```

- The LOW option is an averaging filter. The HIGH option is an edge enhancement filter.

Focal Flow: determines the flow of the values in the surface raster dataset within each cell's immediate neighborhood.

```
FocalFlow <in_surface_raster> <out_raster> {threshold_value}
```



- The resulting values of Focal Flow measure flow into, and not out of, a cell.
- The output values of a function are derived from the binary representation of the results of the analysis.

Focal Statistics: calculates a statistic for each raster dataset cell value within a specified neighborhood.

```
FocalStatistics <in_raster> <out_raster> {neighborhood} {MEAN | MAJORITY | MAXIMUM | MEDIAN
| MINIMUM | MINORITY | RANGE | STD | SUM | VARIETY} {DATA | NODATA}
```

- When a circular, annulus-shaped, or wedge-shaped neighborhood is specified, since the center of the cell must be encompassed within the neighborhood, some of the outer diagonal cells may not be considered in the calculations. However, these cell locations will receive the resulting value from the calculations because they fall within the minimum bounding rectangle (or the output neighborhood) of the three circular neighborhoods.

Line Statistics: calculates a statistic on the attributes of lines in a circular neighborhood around each output cell in a raster dataset.

`LineStatistics <in_polyline_features> <field> <out_raster> {cell_size} {search_radius} {MEAN | MAJORITY | MAXIMUM | MEDIAN | MINIMUM | MINORITY | RANGE | STD | SUM | VARIETY}`

- Only the part of a line within the neighborhood is considered for the Majority, Mean, Median, and Minority types. For the others, it does not matter whether a part or the whole line is used.
- If there are no lines in the neighborhood of a raster cell, then the Variety statistic assigns a value of 0. For the other statistics, NoData is assigned.
- The statistic types Majority, Mean, Median, and Minority are weighted according to the length of the lines. If a line is twice as long as another one, its value is considered to occur twice as often.
- When the field is integer, the available overlay statistic choices are: Mean, Majority, Maximum, Median, Minimum, Minority, Range, STD, and Variety. When the field is floating point, the only allowed statistics are: Mean, Maximum, Minimum, Range, and STD.

Point Statistics: calculates a statistic on the points in a neighborhood outputting a raster dataset.

`PointStatistics <in_point_features> <field> <out_raster> {cell_size} {neighborhood} {MEAN | MAJORITY | MAXIMUM | MEDIAN | MINIMUM | MINORITY | RANGE | STD | SUM | VARIETY}`

- When the field is integer, the available overlay statistic choices are: Mean, Majority, Maximum, Median, Minimum, Minority, Range, STD, Sum, Variety. When the field is floating point, the only allowed statistics are: Mean, Maximum, Minimum, Range, STD, Sum.
- If there are no points in the neighborhood of a raster cell, then the Variety statistic assigns a value of 0. For the other statistics, NoData is assigned.

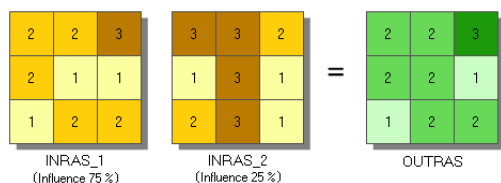


Overlay toolset

Contains the tool to create an output surface by adding a series of weighted input raster datasets together.

Weighted Overlay: overlays several rasters using a common scale and weighing each according to its importance.

`WeightedOverlay <raster{influence} {field} {remap};raster{influence} {field} {remap}...> <out_raster>`



- In the Illustration above, the two input rasters have been reclassified to a common measurement scale of 1 to 3. Each raster is assigned a percentage influence. The cell values are multiplied by their percentage influence, then added to create the output raster. For example, consider the top left cell. The values for the two inputs become $(2 * .75) = 1.5$ and $(3 * .25) = .75$. The sum of 1.5 and .75 is 2.25. Because the output raster from Weighted Overlay is integer, the final value is rounded to 2.
- All input rasters need to be integers. A floating point raster must first be converted to an integer raster before it can be used in Weighted Overlay. The Reclassification tools provide an effective way to do the conversion.
- Each input raster is weighted according to its importance, or percent influence. The weight is a relative percentage, and the sum of the percent influence weights must equal 100 percent.



Raster Creation toolset

Contains tools to create raster datasets based on a constant, random values, or a normal distribution using the existing cell size, extent, and other analysis properties.

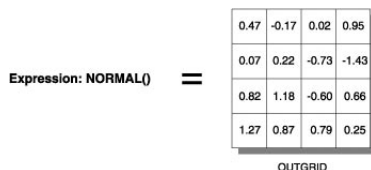
Create Constant Raster: creates a raster dataset from a constant value.

`CreateConstantRaster <out_raster> <constant_value> {INTEGER | FLOAT} {cell_size} {extent}`

- The Constant value must be a numeric value. Scientific notation is accepted (for example, 3.048e-4).

Create Normal Raster: creates a raster dataset of random values from a normal distribution.

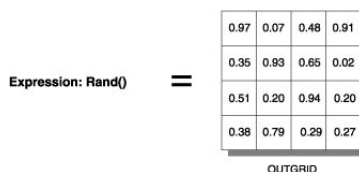
`CreateNormalRaster <out_raster> {cell_size} {extent}`



- The random number generator is automatically seeded with the current value of the system clock (seconds since January 1, 1970). Reseeding the Create Random Raster tool also reseeds Create Normal Raster.
- The output from Create Normal Raster will have a mean of 0 and a standard deviation of 1. If a different standard deviation is desired, multiply the output raster by that value. If a different mean is desired, add that value to the raster. For example, to create a raster in which the values are characterized by a mean of 39 and a standard deviation of 2.5, multiply the results of Create Normal Raster by 2.5, then add 39.

Create Random Raster: creates a raster dataset of random numbers between 0 and 1.

`CreateRandomRaster <out_raster> {seed_value} {cell_size} {extent}`



- Repeatedly using the same seed value, or the default, will not produce the same raster.
- The Random function creates numbers with up to 20 digits after the decimal point.



Reclass toolset

Contains tools to change the values assigned to cells in a thematic raster dataset.

Lookup: creates a new raster dataset by looking up values found in another field in the table of the input raster dataset.

`Lookup <in_raster> <lookup_field> <out_raster>`

- If the lookup field is a numeric type, the values of that field will be written to the output raster attribute table as Value. Other items in the input raster attribute table will not be transferred to the output raster attribute table.

For example, an attribute table of input raster with numeric field Attr1:

Value	Count	Attr1
1	294	1
2	345	8
3	654	3

Output attribute table from Lookup on Attr1 field:

Value	Count
1	294
3	654
8	345

Reclass By ASCII File: reclassifies (or changes) the values of the input cells of a raster dataset by using an ASCII remap file.

`ReclassByASCIIFile <in_raster> <in_remap_file> <out_raster> {DATA | NODATA}`

- The output raster will always be of integer type. If the output assignment values in the ASCII file are floating-point values, an error message will be returned and the program will halt.

Reclass By Table: reclassifies (or changes) the values of the input cells of a raster dataset by using a remap table.

`ReclassByTable <in_raster> <in_remap_table> <from_value_field> <to_value_field>
<output_value_field> <out_raster> {DATA | NODATA}`

- The from value field, to value field, and output value field are the field names in the table that define the remapping.
- To reclassify individual values, use a simple remap table of two items. The first item identifies the value to reclassify, and the other item the value to assign it. Set the to value field to the same as the from value field. The value to assign to the output is output value field.
- To reclassify ranges of values, the remap table must have items defining the start and end of each range, along with the value to assign the range. The item defining the start of the range is the from value field, and the value defining the end of the range is the to value field. The value to assign to the output is output value field.
- The remap table can be an INFO table, a .dbf file, an Access table, or a text file.
- The values in the from and to fields can be any numerical item. The assignment values in the output field must be integers.
- Values in the from field of the table must be sorted in ascending order and should not overlap.

Reclassify: reclassifies (or changes) the value of the cells in a raster dataset.

`Reclassify <in_raster> <reclass_field> <remap> <out_raster> {DATA | NODATA}`

- The remap table can be stored with the Save button. The Load button allows previously created remap tables to be used. Only remap tables created by the tool should be used in Reclassify.
- If running the Reclassify tool as part of a model within a ModelBuilder window, run the tools before the Reclassify tool in the model first. This will allow the values for the input raster to display properly in the Reclassification dialog box.

Slice: slices a range of values of the input cells by zones of equal area or equal interval.

`Slice <in_raster> <out_raster> <number_zones> {EQUAL_INTERVAL | EQUAL_AREA | NATURAL_
BREAKS} {base_output_zone}`

- If a mask has been set, those cells that have been masked out will receive NoData on the output slice raster.

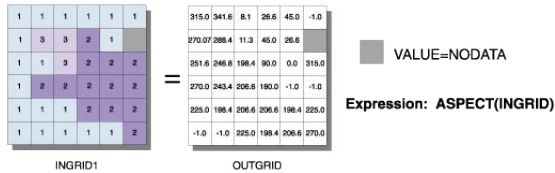


Surface toolset

Contains tools to analyze the surface of the shapes represented by the raster values.

Aspect: identifies the direction of the maximum rate of change in z-value from each cell.

Aspect <in_raster> <out_raster>



- Aspect is the direction of the maximum rate of change in the z-value from each cell in a raster surface.
- Aspect is expressed in positive degrees from 0 to 359.9, measured clockwise from the north.
- Cells in the input raster of zero slope (for example, flat) are assigned an aspect of -1.

Contour: creates contours or isolines from a raster dataset surface.

Contour <in_raster> <out_polyline_features> <contour_interval> {base_contour} {z_factor}

- Smoother but less accurate contours may be obtained by first performing a Neighborhood Focal Statistics operation with the mean option on the input raster.
- A base contour is used, for example, when you want to create contours every 15 meters, starting at 10 meters. Here, 10 would be used for the base contour, and 15 would be the contour interval. The values to be contoured would be 10, 25, 40, 55, and so on.
- Specifying a base contour does not prevent contours from being created above or below that value.

Contour List: creates contours or isolines based on a list of contour values.

ContourList <in_raster> <out_polyline_features> <contour_values;contour_values...>

- Smoother but less accurate contours may be obtained by first performing a Neighborhood Focal Statistics operation with the mean option on the input raster.

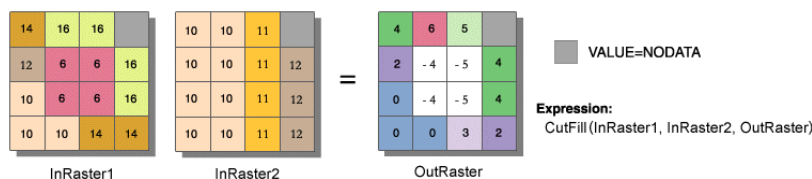
Curvature: calculates the curvature of a surface at each cell center.

Curvature <in_raster> <out_curvature_raster> {z_factor} {out_profile_curve_raster} {out_plan_curve_raster}

- The primary output is the curvature of the surface on a cell-by-cell basis, as fitted through that cell and its eight surrounding neighbors. Curvature is the second derivative of the surface, or the slope of the slope. Two optional output curvature types are possible; the profile curvature is in the direction of the maximum slope, and the plan curvature is perpendicular to the direction of the maximum slope.
- A positive curvature indicates that the surface is upwardly convex at that cell. A negative curvature indicates that the surface is upwardly concave at that cell. A value of zero indicates that the surface is flat.
- In the profile output, a negative value indicates that the surface is upwardly convex at that cell. A positive profile indicates that the surface is upwardly concave at that cell. A value of zero indicates that the surface is flat.
- In the plan output, a positive value indicates that the surface is upwardly convex at that cell. A negative plan indicates that the surface is upwardly concave at that cell. A value of zero indicates that the surface is flat.
- Units of the Curvature output raster, as well as the units for the optional output profile curve raster and output plan curve raster are one over 100 z units, or 1/100 (z units). The reasonably expected values of all three output rasters for a hilly area (moderate relief) may differ from about -0.5 to 0.5; while for the steep, rugged mountains (extreme relief), the values may vary between -4 and 4.

Cut/Fill: calculates cut and fill areas.

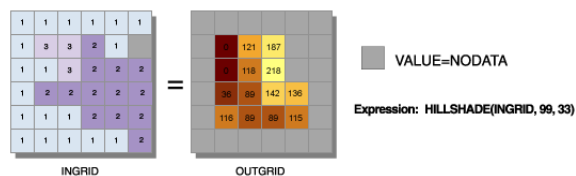
CutFill <in_before_surface> <in_after_surface> <out_raster> {z_factor}



- The Cut/Fill tool enables you to create a map based on two input surfaces (before and after), displaying the areas and volumes of surface materials that have been modified by the addition or removal of surface material. Negative z values indicate regions of the input before raster surface that have been filled; positive regions indicate cuts.
- To get accurate Cut/Fill results, the z units should be the same as the x,y ground units. This ensures that the resulting volumes are meaningful cubic measures (for example, cubic meters). If they are not the same, use a z-factor to convert z units to x,y units. For example, if your x,y units are meters and your z units are feet, you could specify a z-factor of 0.3048 to convert feet to meters.

Hillshade: creates a shaded relief raster by considering the illumination angle and shadows.

Hillshade <in_raster> <out_raster> {azimuth} {altitude} {NO_SHADOWS | SHADOWS} {z_factor}



- The Hillshade tool creates a shaded relief raster from a raster. The illumination source is considered at infinity.
- Two types of shaded relief rasters can be output. Having model shadows unchecked outputs a raster that only considers the local illumination angle. Having model shadows checked outputs one that considers the effects of both the local illumination angle and shadow.

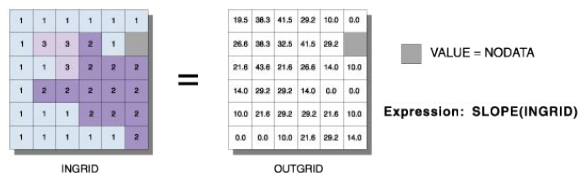
Observer Points: identifies exactly which observer points are visible from each surface location.

ObserverPoints <in_raster> <in_observer_point_features> <out_raster> {z_factor} {FLAT_EARTH | CURVED_EARTH} {refractivity_coefficient}

- The visibility of each cell center is determined by comparing the altitude angle to the cell center with the altitude angle to the local horizon. The local horizon is computed by considering the intervening terrain between the point of observation and the current cell center. If the point lies above the local horizon, it is considered visible.

Slope: identifies the rate of maximum change in z-value from each cell.

Slope <in_raster> <out_raster> {DEGREE | PERCENT_RISE} {z_factor}



- Slope is the rate of maximum change in z-value from each cell.
- The use of a z-factor is essential for correct slope calculations when the surface z units are expressed in units different from the ground x,y units.
- Degree of slope is a value between 0 and 90.

Viewshed: derives the raster dataset surface locations visible to a set of observation points.

`Viewshed <in_raster> <in_observer_features> <out_raster> {z_factor} {FLAT_EARTH | CURVED_EARTH} {refractivity_coefficient}`

- The visibility of each cell center is determined by comparing the altitude angle to the cell center with the altitude angle to the local horizon. The local horizon is computed by considering the intervening terrain between the point of observation and the current cell center. If the point lies above the local horizon, it is considered visible.



Zonal toolset

Contains tools that can be applied to the zones within the raster dataset.

Tabulate Area: calculates cross-tabulated areas between two datasets.

`TabulateArea <in_zone_data> <zone_field> <in_class_data> <class_field> <out_table> {processing_cell_size}`

- Zones are created by identical integer values. If the input is a raster, it must be integer. If the input is a feature class, the Zone field must be integer.
- There will be a field in the output table for each unique value of the input raster or feature zone data.
- There will be a record in the output table for each unique value of the input raster or feature class data.
- Each record in the output table will store the area of each class within each zone.
- If a point or line dataset is used as the class data, then the area intersected by those features will be reported.

Zonal Fill: fills zones in a dataset using the minimum cell value from another raster dataset along the zone boundary.

`ZonalFill <in_zone_raster> <in_weight_raster> <out_raster>`

- Zonal Fill can be used to fill sinks to the minimum elevation of their watershed boundary. This is a step in determining the depth of sinks before filling to create a depressionless digital elevation model (DEM).
- The input zone raster must be integer.

Zonal Geometry: calculates area, perimeter, thickness, or the characteristics of ellipse and records the information as a raster dataset.

`ZonalGeometry <in_zone_data> <zone_field> <out_raster> {AREA | PERIMETER | THICKNESS | CENTROID} {cell_size}`

- Zones are created by identical integer values. If the input is a raster, it must be integer. If the input is a feature class, the zone field must be integer.
- If the Zone dataset is a feature dataset, it is recommended that you ensure the extent and snap rasters are set appropriately in the environment settings or raster settings. This will ensure that the results of the vector-to-raster conversion will align properly with the Value raster.

Zonal Geometry As Table: calculates area, perimeter, thickness, or the characteristics of ellipse and records the information in an output table.

`ZonalGeometryAsTable <in_zone_data> <zone_field> <out_table> {processing_cell_size}`

- Zones are created by identical integer values. If the zone input is a raster, it must be integer. If the zone input is a feature class, the zone field must be integer.
- If the input raster or feature zone data is a feature dataset, a cell size must be set in either the dialog box or the Environment. To align the feature dataset to the value raster, it is recommended that the extent and snap rasters are set appropriately in the environment settings and raster settings.

- All the results in the output table are presented in map units. Only the values of the ORIENTATION item are in degrees, with a possible range of 0 to 180. The ORIENTATION is defined as an angle between the x axis and the major axis of the ellipse. The values of the orientation angle increase counterclockwise, starting from 0 in the east (horizontal, to the right) and going through 90 when the major axis is vertical.
- Following is an example of Zonal Geometry As Table on the following input:



The contents of the output table would be:

VALUE	AREA	PERIMETER	THICKNESS	XCENTROID	YCENTROID	MAJORAXIS	MINORAXIS	ORIENTATION
0	5.0	14.0	0.5	2.300	2.100	2.338	0.681	60.710
1	5.0	14.0	0.5	1.900	2.100	2.668	0.596	126.061
2	6.0	8.0	0.5	3.167	2.167	1.286	0.743	130.00
4	2.0	6.0	0.5	0.500	1.000	1.128	0.564	90.00

Zonal Statistics: calculates a statistic on values of a raster within the zones of another dataset.

`ZonalStatistics <in_zone_data> <zone_field> <in_value_raster> <out_raster>`
 {MEAN | MAJORITY | MAXIMUM | MEDIAN | MINIMUM | MINORITY | RANGE | STD | SUM | VARIETY}
 {DATA | NODATA}

- The input value raster can be either integer or floating point. However, when it is of floating-point type, the zonal calculations for majority, median, minority, and variety are not available.
- For Majority and Minority, when there is a tie, the output for all cell locations in the zone are assigned the lowest of the tied values.

Zonal Statistics As Table: summarizes values of a raster dataset within the zones of another dataset and reports the results to a table.

`ZonalStatisticsAsTable <in_zone_data> <zone_field> <in_value_raster> <out_table>`
 {DATA | NODATA}

- The input value raster can be either integer or floating point. However, when it is of floating-point type, the zonal calculations for majority, median, minority, and variety are not available.
- For majority and minority calculations, when there is a tie, the output for all cell locations in the zone are assigned the lowest of the tied values.
- The number of rows in the output table is the number of zones.
- An example of the output table is:

VALUE	COUNT	AREA	MEAN	MIN	MAX	...
0	5	125.0000	0.6	0.0	0.0	...
1	5	125.0000	1.0	0.0	3.0	...
2	3	75.0000	1.667	1.0	2.0	...
4	2	50.0000	3.0	3.0	3.0	...

Index

3D Analyst toolbox 79

A

Absolute value. *See* Abs command
 Abs command 122
 ACosH command 134
 ACos command 134
 AddCADFields command 14
 AddCodedValueToDomain command 40
 AddFeatureClassToTopology command 62
 AddFieldToAnalysisLayer command 97
 AddField command 46
 AddIndex command 50
 Adding. *See* Plus command; Plus command
 attribute index 50
 CAD fields 14
 coded value to domain 40
 fields 46
 indexes 50
 items 33
 joins 51
 rules to topology 62
 spatial indexes 50
 subtypes 60
 x,y coordinates 35, 43
 AddItem command 33
 AddJoin command 51
 AddLocations command 97
 Address
 geocoding 67
 locator
 creating 67
 deleting 67
 standardizing 68
 AddressBuild. *See* RebuildGeocodingIndex command
 AddressCreate. *See* CreateAddressLocator command
 AddressMatch. *See* GeocodeAddresses command
 AddressParse. *See* StandardizeAddresses command
 AddRouteMeasure. *See* LocateFeaturesAlongRoutes
 command
 AddRuleToTopology command 62
 AddSpatialIndex command 50
 AddSubtype command 60
 AddXY command 35, 43
 Adjust. *See* Transform command
 ADRGGrid. *See* CopyRaster command
 Advanced TIGER Conversion. *See* TigerTool command
 Aggregate
 command 108
 toolset 28
 AggregatePolygons command 30
 Aggregate Polygons. *See* AreaAggregate command

Alias
 described 1
 list 2
 AlterVersion command 63
 Analysis
 toolbox 5
 toolset 21, 97
 Analyze command 61
 Analyzing
 clusters and outliers 72
 hot spots 72, 73
 Analyzing Patterns toolset 71
 And
 bitwise 127
 Boolean 129
 combinatorial 130
 Annotation
 importing from CAD 16
 importing from coverage 16
 AppendAnnotation command 41
 Append command 28, 47
 ArcDLG command 25
 ArcDXF. *See* ExportCAD command
 ArcIGDS. *See* ExportCAD command
 ArcInfo workspace
 creating 65
 ArcRoute command 29
 ArcS57 command 26
 ArcSDE
 checking in 39
 checking out 39
 ArcSection. *See* CreateRoutes command
 ArcShape. *See* FeatureClassToShapefile command
 ArcTiger. *See* TigerArc command; TigerTool command
 ArcTIN. *See* Editing: TINs
 Area
 calculating 75
 creating polygon features 45
 AreaAggregate. *See* AggregatePolygons command
 Arguments 2
 ASCII
 converting to raster 13
 ASCIIGrid. *See* ASCII: converting to raster
 ASCIIToRaster command 17
 ASinH command 135
 ASin command 134
 Aspect command 87, 144
 AssignDefaultToField command 47
 AssignDomainToField command 40
 Assigning
 domain to field 40
 ATan2 command 135
 ATanH command 136
 ATan command 135
 Attributes
 joining tables 33

- Autocorrelation 71
- AutomateGeocodingIndexes command 67
- AverageNearestNeighbor command 71
- B**
- BandCollectionStats command 137
- Band Collection Statistics. *See* BandCollectionStats command
- Base 10. *See* Exp10 command; Log10 command
- Base 2. *See* Exp2 command; Log2 command
- Base e. *See* Exp command
- Basic TIGER Conversion; TigerArc command
- Basin command 113
- BatchBuildPyramids command 56
- BatchCalculateStatistics command 56
- BatchProject command 53
- Bitwise
 - And 127
 - complement. *See* BitwiseNot command
 - Exclusive Or 128. *See also* BitwiseXOr command
 - left shift 127
 - Not 128
 - Or 128
 - right shift 128
 - toolset 127
- BitwiseAnd command 127
- BitwiseLeftShift command 127
- BitwiseNot command 128
- BitwiseOr command 128
- BitwiseRightShift command 128
- BitwiseXOr command 128
- Bitwise Complement. *See* BitwiseNot command
- Bitwise Exclusive Or. *See* BitwiseXOr command
- BlockMajority. *See* BlockStatistics command
- BlockMax. *See* BlockStatistics command
- BlockMean. *See* BlockStatistics command
- BlockMedian. *See* BlockStatistics command
- BlockMin. *See* BlockStatistics command
- BlockMinority. *See* BlockStatistics command
- BlockRange. *See* BlockStatistics command
- BlockStatistics command 140
- BlockStd. *See* BlockStatistics command
- BlockSum. *See* BlockStatistics command
- BlockVariety. *See* BlockStatistics command
- Boolean
 - and 129
 - complement. *See* BooleanNot command
 - exclusive or 130
 - not 129
 - or 130
- BooleanAnd command 129
- BooleanNot command 129
- BooleanOr command 130
- BooleanXOr command 130
- Boolean Complement. *See* BooleanNot command
- BoundaryClean command 109
- Buffer
 - command 7
 - using multiple rings 8
- Buffer command 23
- Building
 - pyramids 56
 - simplification 32
- BuildNetwork command 98
- BuildPyramids command 56
- BuildSta. *See* CellStatistics command
- Build command 36
- C**
- CAD
 - annotation
 - importing 16
 - importing 16
- CalculateAreas command 75
- CalculateDefaultClusterTolerance command 42
- CalculateDefaultGridIndex command 42
- CalculateField command 47
- CalculateLocationFields command 97
- CalculateStatistics command 56
- Calculating
 - cluster tolerance 42
 - fields 47
 - kernel density 102
 - spatial grid index 50
 - statistics 56
 - summary statistics 9
- CalibrateRoutes command 69
- CAnd. *See* CombinatorialAnd command
- Cartography toolbox 11
- CellStatistics command 119
- Cell groupings
 - using isodata clustering 139
- Centerline
 - creating 30
- CentralFeature command 73
- CentroidLabels. *See* FeatureToPoint command
- ChangePrivileges command 61
- CheckGeometry command 43
- CheckInDelta command 39
- Checking in
 - from Delta geodatabases 39
 - to ArcSDE 39
- Checking out
 - from ArcSDE 39
- CheckIn command 39
- CheckOut command 39
- ClassProb. *See* ClassProbability command
- ClassProbability command 137
- ClassSig. *See* CreateSignatures command
- Cleaning
 - boundaries 109

Clean command	36	CopyRows command	61
Clip command	5, 21, 57	Copy command	47
Clustering		COr. <i>See</i> CombinatorialOr command	
High/Low	71	Corridor command	103
ClustersOutliersRendered command	72	CosH command	136
ClustersOutliers command	72	Cost	
Cluster and outlier		calculating least accumulative distance	104
analysis	72	least accumulative	104
Cluster tolerance		least accumulative distance	
calculating	42	path	104, 106
setting	63	least accumulative path	
CollapseDualLineToCenterline command	30	define neighbor cell	103
CollectEventsRendered command	75	least accumulative source	106
CollectEvents command	75	CostAllocation command	103
Combinatorial		CostBackLink command	103
and	130	CostDistance command	104
exclusive or	131	CostPath command	104
or	130	Cos command	136
CombinatorialAnd command	130	Counting	
CombinatorialOr command	130	table rows. <i>See</i> GetCount command	
CombinatorialXOr command	131	CountRenderer command	75
Combine command	119	Coverages	
Combining features. <i>See</i> Append command		appending	28
Compact command	39	combining	28
CompositeBands command	57	creating	37
Composite Features toolset	28	erasing	22
Compress command	39	exporting as text file	26
Conditional toolset	101	exporting to DLG	25
Conflicts		generating	27
finding	31	generating topology	36
Contouring		importing from DLG	27
from a TIN	91	importing from S57	27
ContourList command	88, 144	importing from SDTS	28
Contour command	88, 144	importing from VPF	28
Conversion		projecting	34
toolbox	13	setting tolerances	36
toolset	25, 79	toolbox	21
ConvertImage. <i>See</i> CopyRaster command		transforming	34
Converting		updating	23
to feature class	15	updating attribute table and topology	36
to shapefile	15	CreateAddressLocator command	67
ConvertRemap. <i>See</i> ReclassByASCIIFile command		CreateArcInfoWorkspace command	65
Con command	101	CreateCADXData command	14
Coordinates		CreateConstantRaster command	142
adding x,y	35	CreateDomain command	40
CopyFeatures command	43	CreateFeatureClass command	42
Copying		CreateFeatureDataset command	65
features	43	CreateFolder command	65
feature classes	47	CreateLabels command	37
feature datasets	47	CreateNormalRaster command	142
raster catalog items	58	CreatePersonalGDB command	65
raster datasets	57	CreateRandomRaster command	142
rows in a table	61	CreateRasterCatalog command	58
tables	47	CreateRasterDataset command	58
CopyRasterCatalogItems command	58	CreateRelationshipClass command	60
CopyRaster command	57	CreateRoutes command	69

- CreateSignatures command 138
- CreateSpatialReference command 53
- CreateTable command 61
- CreateTIN command 90
- CreateTopology command 63
- CreateTurnFeatureClass command 99
- CreateVersion command 64
- CreateWorkspace. *See* CreateArcInfoWorkspace command
- Create command 37
- Creating
 - address locators 67
 - ArcInfo workspaces 65
 - CAD x data 14
 - centerlines 30
 - coverages 37
 - domains 40
 - feature classes 42
 - feature datasets 65
 - folders 65
 - layers 51
 - multiband raster datasets 57
 - personal geodatabases 65
 - raster catalogs 58
 - raster datasets 58, 142
 - relationship classes 60
 - spatial references 53
 - tables 61
 - tables from domains 41
 - table to relationship classes 60
 - TINs 90
 - topology 63
 - versions 64
- CuldeSacMasks command 11
- Curvature command 88, 144
- CutFill command 88, 145
- CXOr. *See* CombinatorialXOr command
- D**
- DarcyFlow command 111
- DarcyVelocity command 112
- Data
 - selecting 48
- Database toolset 39
- Datasets
 - copying 47
 - deleting 47, 48
 - renaming 48
- Data Interoperability toolbox 93
- Data Management toolbox 39
- Data Management toolset 28
- DBaseInfo. *See* TableToGeodatabase command
- DBMSInfo. *See* TableToGeodatabase command
- DeautomateGeocodingIndexes command 67
- DefineProjection command 34, 53
- DeleteAddressLocator command 67
- DeleteCodedValueFromDomain command 40
- DeleteDomain command 41
- DeleteFeatures command 44
- DeleteField command 47
- DeleteRasterCatalogItems command 58
- DeleteRows command 61
- DeleteVersion command 64
- Delete command 47
- Deleting
 - address locators 67
 - coded values from domains 40
 - domains 41
 - features 44
 - feature classes 47, 48
 - feature datasets 47, 48
 - fields in a table 47
 - geodatabase versions 64
 - items in a raster catalog 58
 - raster datasets 47, 48
 - rows in a table 61
 - tables 47, 48
- Delta geodatabase
 - checking in 39
 - checking out 40
- DEM
 - converting to raster 117
- DEMGrid. *See* DEM: converting to raster
- DEMLattice. *See* DEM: converting to raster
- DEMToraster command 18
- Dendrogram command 138
- Density
 - calculating kernel 102
 - calculating using line features 102
 - calculating using point features 102
 - toolset 102
- DFADArc. *See* Feature class: converting to coverage
- DirectionalDistribution command 73
- DirectionalMean command 74
- Directions command 97
- Disconnected Editing toolset 39
- DissolveRouteEvents command 69
- Dissolve command 31, 49
- Dissolving
 - route events 69
- Distance toolset 103
- Divide command 85, 122. *See also* Mod command
- DLG
 - importing to coverage 27
- DLGArc command 27
- Domain
 - creating 40
 - creating from table 41
 - deleting 41
 - removing from field 41
 - setting value for range 41
- Domains toolset 40

DomainToTable command	41
Drainage. <i>See</i> Hydrology toolset	
DropIndex command	33. <i>See also</i> RemoveIndex command
DropItem command	33
Dropping	
indexes	33
items	33
DTEDGrid. <i>See</i> CopyRaster command	
DXFArc. <i>See</i> ImportCAD command	
DXFINFO. <i>See</i> ImportCAD command	
E	
Editing	
TINs	90
EditSig. <i>See</i> EditSignatures command	
EditSignatures command	138
EditTin command	90
Eliminate command	31, 49
Ellipse	2
Envelopes	
creating polygons from	44
EqualityTest. <i>See</i> EqualTo command	
EqualToFrequency command	119
EqualTo command	131
Erase command	6, 22
ETAKArc. <i>See</i> FeatureClassToCoverage command	
EucAllocation command	105
EucDirection command	105
EucDistance command	105
Euclidean	
allocating	105
direction	105. <i>See also</i> EucDirection command
distance	105. <i>See also</i> EucDistance command
Euclidean Allocation. <i>See</i> EucAllocation command	
Euclidean Direction. <i>See</i> EucDirection command	
Euclidean Distance. <i>See</i> EucDistance command	
EventArc. <i>See</i> MakeRouteEventLayer command	
EventPoint. <i>See</i> MakeRouteEventLayer command	
Events	
dissolving for routes	69
transforming routes	70
EventTransform. <i>See</i> TransformRouteEvents command	
Example tool	2
Exclusive arguments	2
Exclusive or	
bitwise	128
Boolean	130
combinatorial	131
Exp10 command	123
Exp2 command	123
Expand command	109
Exponential. <i>See also</i> Exp command	
base 10	123
base 2	123
base e	122
ExportCAD command	14

Exporting	
feature attribute to ASCII	75
to CAD	14
to Delta geodatabase	40
to Interchange file	25
to S57	26
to SDTS	26
to VPF	26
ExportToDelta command	40
ExportXYv command	75
Export command	25
Exp command	122
ExtractByAttributes command	107
ExtractByCircle command	107
ExtractByMask command	107
ExtractByPoints command	107
ExtractByPolygon command	108
ExtractByRectangle command	108
Extraction toolset	107
ExtractValuesToPoints command	108
Extract toolset	5, 21
F	
FeatureClassToCoverage command	15
FeatureClassToFeatureClass command	15
FeatureClassToGeodatabase command	16
FeatureClassToShapefile command	19
FeatureEnvelopeToPolygon command	44
FeatureOutlineMasks command	11
Features	
buffering	7
center of concentration	74
centrally located	73
concentration or dispersion	74
converting to raster	18
copying	43
deleting	44
dissolving	49
eliminating	49
erasing	6
extracting	5
locating along routes	69
splitting	5
toolset	43
unioning	7
FeatureToLine command	44
FeatureToPoint command	44
FeatureToPolygon command	45
FeatureToRaster command	18
FeatureVerticesToPoints command	45
Feature class	
combining	47
converting to coverage	15
converting to feature class	15
converting to shapefile	15
copy to geodatabase	15

- Feature class (continued)
 - creating..... 42
 - intersecting..... 6
 - remove from topology..... 63
 - toolset..... 41
 - updating attributes and geometry..... 7
- Feature toolset..... 53
- Fields
 - adding..... 46
 - assigning default to..... 47
 - calculating..... 47
 - deleting..... 47
 - toolset..... 46
- Fill command..... 113
- Filter command..... 140
- FindConflicts command..... 31
- Flip command..... 54
- Float
 - command..... 85, 123
 - convert to raster..... 142
- FloatGrid. *See* FloatToRaster command
- FloatToRaster command..... 18
- Flow. *See* DarcyFlow command
- FlowAccumulation command..... 114
- FlowDirection command..... 114
- FlowLength command..... 114
- FMod. *See* Mod command
- FocalFlow command..... 140
- FocalMajority. *See* FocalStatistics command
- FocalMax. *See* FocalStatistics command
- FocalMean. *See* FocalStatistics command
- FocalMedian. *See* FocalStatistics command
- FocalMin. *See* FocalStatistics command
- FocalMinority. *See* FocalStatistics command
- FocalRange. *See* FocalStatistics command
- FocalStatistics command..... 140
- FocalStd. *See* FocalStatistics command
- FocalSum. *See* FocalStatistics command
- FocalVariety. *See* FocalStatistics command
- Folder
 - creating..... 65
- Frequency command..... 8
- From Coverage toolset..... 25
- From Raster toolset..... 13
- Functional Surface toolset..... 81
- G**
 - GALayerToGrid command..... 95
 - Generalization toolset..... 30, 49, 108
 - Generalize. *See* Lines: simplifying
 - General toolset..... 47
 - Generate command..... 27
 - GeocodeAddresses command..... 68
 - Geocoding
 - addresses..... 67
 - deautomating indexes..... 67
 - index
 - rebuilding..... 68
 - toolbox..... 67
 - Geodatabase
 - creating..... 65
 - Geometries
 - checking..... 43
 - repairing..... 46
 - Geostatistical Analyst toolbox..... 95
 - GetCount command..... 62
 - Getis-Ord General G..... 71
 - Getis-Ord Gi*..... 72, 73
 - GreaterThanEqual command..... 132
 - GreaterThanFrequency command..... 120
 - GreaterThan command..... 131
 - GridASCII. *See* RasterToASCII command
 - GridClip. *See* ExtractByRectangle command
 - GridComposite. *See* CompositeBands command
 - GridFlip. *See* Flip command
 - GridFloat. *See* RasterToFloat command
 - GridImage. *See* CopyRaster command
 - GridLine. *See* RasterToPolyline command
 - GridLineShape. *See* RasterToPolyline command
 - GridMirror. *See* Mirror command
 - GridPoint. *See* RasterToPoint command
 - GridPointShape. *See* RasterToPoint command
 - GridPoly. *See* RasterToPolygon command
 - GridRotate. *See* Rotate command
 - GridShape. *See* RasterToPolygon command
 - GridShift. *See* Shift command
 - GridWarp. *See* Warp command
 - Groundwater
 - calculating steady flow..... 111
 - dispersion of point..... 113
 - toolset..... 111
- H**
 - HighestPosition command..... 120
 - HighLowClustering command..... 71
 - Hillshade command..... 89, 145
 - HotSpotsRendered command..... 73
 - HotSpot command..... 72
 - Hydrodynamic dispersion. *See* PorousPuff command
 - Hydrology toolset..... 113
- I**
 - IDEdit command..... 35
 - Identity command..... 6, 22
 - IDs
 - updating..... 99
 - IDW command..... 82, 116
 - IGDSArc. *See* ImportCAD command
 - IGDSInfo. *See* ImportCAD command

ImageGrid. *See* CopyRaster command
 ImportCADAnnotations command 16
 ImportCAD command 16
 ImportCoverageAnnotations command 16
 Importing
 CAD annotation 16
 coverage annotation 16
 from CAD 16
 from Interchange file 27
 from S57 27
 from SDTS 28
 from VPF 28
 Import command 27
 IncreaseMaximumEdges command 99
 Index
 adding 50
 adding spatial index 50
 automating geocoding index 67
 dropping 33
 item 33
 rebuilding for geocoding 68
 removing 50
 Indexes toolset 33, 50
 IndexItem command 33. *See also* AddIndex command
 Info
 joining tables 33
 InfoDBASE. *See* TableToDBASE command
 Integer
 converting from. *See* Int command
 Integrate command 42
 Interoperability tools 93
 InterpolatePolyToPatch command 90
 InterpolateShape command 81
 Interpolating
 raster datasets 82, 116
 raster surfaces 82, 116
 z-values 81
 Interpolation
 calculating the trend on a point dataset 119
 toolset 116
 Intersecting
 coverages 22, 23
 IntersectingLayersMasks command 11
 Intersect command 6, 22
 Int command 85, 123
 Inverse distance weighted 82, 116
 IsNull command 132
 IsoCluster command 139
 Item
 adding 33
 dropping 33
 indexing 33
 Items toolset 33

J

Joining
 attribute tables 51
 info tables 33
 JoinItem command 33
 Joins
 adding 51
 removing 51
 toolset 33, 51

K

KernelDensity command 102
 Kill. *See* Delete command
 Kriging command 82, 116

L

LatticeClip. *See* ExtractByPolygon command
 LatticeContour. *See* Contour command
 Layer
 make from feature 51
 make from query table 51
 make from raster 52
 make from raster catalog 52
 make from x,y events 52
 Layers and Table Views toolset 51
 LessThanEqual command 132
 LessThanFrequency command 120
 LessThan command 132
 Linear directional mean 74
 Linear Referencing toolbox 69
 LineDensity command 102
 LineGrid. *See* FeatureToRaster command
 LineOfSight command 81
 Lines
 converting
 coverage to region 28
 coverage to route 29
 creating from polygon 45
 generating for polygon 46
 simplifying 32, 49
 smoothing 49
 splitting at vertices 46
 LineStatistics command 141
 LineStats. *See* LineStatistics command
 Ln command 124
 Local toolset 119
 LocateFeaturesAlongRoutes command 69
 Locating
 features along routes 69
 Log10 command 124
 Log2 command 124
 Logarithm. *See also* Ln command
 base 10 124
 base 2 124
 base e 124
 natural 124

- Logical toolset 129
- Lookup command 86, 142
- LowestPosition command 120
- M**
- Majority. *See* CellStatistics command
- MajorityFilter command 109
- MakeClosestFacilityLayer command 97
- MakeFeatureLayer command 51
- MakeODCostMatrixLayer command 98
- MakeQueryTable command 51
- MakeRasterCatalogLayer command 52
- MakeRasterLayer command 52
- MakeRouteEventLayer command 70
- MakeRouteLayer command 98
- MakeServiceAreaLayer command 98
- MakeTableView command 52
- MakeXYEventLayer command 52
- Mapping Clusters toolset 72
- Map Algebra
 - statements 121
 - toolset 121
- Mask
 - creating cul-de-sac 11
 - creating from feature outline 11
- Masking toolset 11
- MasksFromFeatureOutlineMasks command 11
- MasksFromIntersectingLayersMasks command 11
- Math toolset 122
- Max. *See* CellStatistics command
- Maximum Likelihood Classification. *See* MLClassify command
- Mean. *See* CellStatistics command
- MeanCenter command 74
- MeasureRoute. *See* CreateRoutes command
- Measuring Geographic Distributions toolset 73
- Med. *See* CellStatistics command
- Min. *See* CellStatistics command
- Minimum curvature. *See* Spline command
- Minority. *See* CellStatistics command
- Minus command 86, 124
- Mirror command 54
- MLClassify command 139
- Model
 - described 1
- Mod command 125
- Moran's I 71
- MosaicToNewRaster command 59
- Mosaic command 58
- Multiband raster dataset
 - creating 57
- MultiOutputMapAlgebra command 121
- MultipartToSinglepart command 45
- Multipart feature
 - converting to single-part feature 45
- MultipleRingBuffer command 8
- Multiply. *See* Times command
- Multivariate toolset 137
- N**
- NaturalNeighbor command 82, 117
- Nearest neighbor 71
- Near command 8, 24
- Negate command 125
- Neighborhood toolset 140
- Network Analyst toolbox 97
- Network Dataset toolset 98
- Nibble command 110
- Nodes
 - renumber 35
- Normal. *See* CreateNormalRaster command
- Not
 - bitwise 128
 - Boolean 129
- NotEqual command 133
- O**
- ObserverPoints command 89, 145
- Or
 - bitwise 128
 - Boolean 130
 - combinatorial 130
- Outlier and cluster analysis 72
- Overlaying
 - route events 70
- OverlayRouteEvents command 70
- Overlay toolset 6, 22, 141
- P**
- Parameters 2
 - optional 2
 - required 2
- Particle
 - calculating path 112
- ParticleTrack command 112
- PathAllocation command 106
- PathBackLink command 106
- PathDistance command 106
- Pick command 101
- PivotTable command 62
- Plus command 86, 125
- Point
 - calculating at surface. *See* SurfaceSpot command
- PointDensity command 102
- PointDistance command 8, 24
- PointGrid. *See* FeatureToRaster command
- PointNode command 24
- Points
 - calculating at surface. *See* SurfaceSpot command
 - calculating distance between 8, 24
 - calculating distance from 8, 24

- Points (continued)
 - creating from features 44
 - creating from vertices 45
 - transferring attributes to node features 24
- PointStatistics command 141
- Polygons
 - aggregating 30
 - converting coverages to regions 29
 - converting coverage to region 29
 - converting to lines 46
 - dissolving 31
 - merging 31
- PolygonToLine command 46
- PolyGrid. *See* FeatureToRaster command
- PolyRegion command 29
- Popularity command 121
- PopulateAlternateIDFields command 99
- PorousPuff command 113
- PostVersion command 64
- Power command 125
- PrincipalComponents command 140
- PrinComp. *See* PrincipalComponents command
- Privileges
 - changing 61
- ProjectDefine. *See* DefineProjection command
- ProjectGrid. *See* ProjectRaster command
- Projecting
 - rasters 54
- Projection
 - defining 34, 53
- Projections and Transformations toolset 53
- Projections toolset 34
- ProjectRaster command 54
- Project command 34, 53
- Proximity toolset 7, 23
- Pyramids
 - building 56
- Q**
 - QuickExport command 93
 - QuickImport command 93
- R**
 - Range. *See* CellStatistics command
 - Rank command 121
 - RasterTIN command 79
 - RasterToASCII command 13
 - RasterToFloat command 13
 - RasterToGeodatabase command 17
 - RasterToOtherFormat command 18
 - RasterToPoint command 13
 - RasterToPolygon command 13
 - RasterToPolyline command 13
 - Raster catalogs
 - copying items 58
 - creating 58
 - deleting items 58
 - making layers 52
 - Raster Creation toolset 142
 - Raster datasets
 - calculating accumulated flow 114
 - calculating aspect 13
 - calculating band statistics 56
 - calculating drainage basins 113
 - calculating flow direction 114
 - calculating flow distance 114
 - calculating slope 89, 145
 - calculating statistics 56, 140
 - calculating surface length 81
 - calculating surface values 81
 - calculating surface visibility 81
 - calculating surface volume 81
 - calculating weighted overlay 141
 - changing cell values. *See* Reclassify command
 - changing scale 54
 - clipping 57
 - combining 119
 - converting from TINs 80
 - converting to ASCII 13
 - converting to linear network features 115
 - converting to points 13
 - converting to polygons 13
 - converting to polylines 13
 - converting values to float 13, 123
 - converting values to integer. *See* Int command
 - creating 58, 82, 85, 142
 - creating ASCII file of selected cells 108
 - creating contours 88, 144
 - creating surface. *See* Raster Interpolation toolset
 - creating TINs 90
 - creating using elevation data 117
 - extracting cells
 - using attributes 107
 - using a mask 107
 - using points 107
 - within a circle 107
 - within a polygon 108
 - within a rectangle 108
 - extracting cell values
 - using points 107
 - interpolating 82, 90, 116
 - majority filter 109
 - making layers 52
 - mosaicking 58, 59
 - performing a classification 139, 140
 - principal components 140
 - projecting 54
 - reclassifying 86, 87, 143
 - recording least cost path 104

Raster datasets (continued)	
reducing resolution.....	108
replacing cell values. <i>See</i> Nibble command	
resampling.....	59
rotating.....	55
surface	
calculating viewable areas.....	89, 146
removing sinks.....	113
surface flow values.....	140
transforming.....	54
viewshed.....	89, 146
Raster Interpolation toolset.....	82
Raster Math toolset.....	85
Raster Reclass toolset.....	86
Raster Surface toolset.....	87
Raster toolset.....	54, 56
RebuildGeocodingIndex command.....	68
Rebuilding	
geocoding indexes.....	68
Reclass. <i>See</i> ReclassByTable command	
toolset.....	142
ReclassByASCIIFile command.....	87, 143
ReclassByTable command.....	87, 143
Reclassify command.....	87, 143
ReconcileVersion command.....	64
Reconciling	
versions.....	64
Rectify. <i>See</i> ProjectRaster command	
Reducing database sizes.....	39
RegionClass command.....	28
RegionGroup command.....	110
RegionPoly command.....	29
Regions	
converting to polygon coverages.....	29
grouping.....	110
RegisterAsVersioned command.....	64
Relationship Classes toolset.....	60
RemoveDomainFromField command.....	41
RemoveFeatureClassFromTopology command.....	63
RemoveIndex command.....	50
RemoveJoin command.....	51
RemoveRuleFromTopology command.....	63
RemoveSpatialIndex command.....	50
RemoveSubtype command.....	60
Removing	
attribute indexes.....	50
feature classes from topologies.....	63
joins.....	51
rules from topologies.....	63
spatial indexes.....	50
Rename command.....	48
Rendering	
and collecting events.....	75
counts.....	75
with hot spot analysis.....	73
Renode command.....	35
Renumbering	
nodes.....	35
RepairGeometry command.....	46
Resample command.....	59
Rescale command.....	54
Reselect command.....	21. <i>See also</i> Select command
Rotate command.....	55
RoundDown command.....	125
RoundUp command.....	126
Routes	
creating from line coverages.....	29
locating features along.....	69
overlaying events.....	70
transforming events.....	70
Rows	
copying.....	61
counting.....	62
deleting.....	61
Rubber sheeting rasters.....	55
Rules	
adding to topology.....	62
removing from topology.....	63
S	
S57Arc command.....	27
Sample command.....	108
SaveToLayerFile command.....	52
Saving	
layer files.....	52
Scale	
changing.....	54
Script	
described.....	1
SDTSExport command.....	26
SDTSImport command.....	28
SelectBox. <i>See</i> ExtractByRectangle command	
SelectCircle. <i>See</i> ExtractByCircle command	
SelectData command.....	48
Selecting	
by attributes.....	52
by locations.....	52
data.....	48
layers by attributes.....	52
layers by locations.....	52
SelectLayerByAttribute command.....	52
SelectLayerByLocation command.....	52
SelectMask. <i>See</i> ExtractByMask command	
SelectPoint. <i>See</i> ExtractByPoints command	
SelectPolygon command. <i>See</i> ExtractByPolygon command	
Select command.....	5
SetCADAlias command.....	14
SetClusterTolerance command.....	63
SetDefaultSubtype command.....	61
SetNull command.....	101
SetSubtypeField command.....	61

- Setting
 - CAD aliases..... 14
 - SetValueForRangeDomain command..... 41
 - Shaded relief. *See* Hillshade command
 - ShapeGrid. *See* FeatureToRaster command
 - Shift command..... 55
 - Shrink command..... 110
 - Signature files
 - creating..... 138
 - creating probability layers..... 137
 - editing..... 138
 - SimplifyBuilding command..... 32
 - Simplifying
 - buildings..... 32
 - lines..... 32
 - SimplifyLineOrPolygon command..... 32
 - SimplifyLine command..... 49
 - SingleOutputMapAlgebra command..... 121
 - SinH command..... 137
 - Sink command..... 115
 - Sin command..... 136
 - Slice command..... 87, 143
 - Slope command..... 89, 145
 - SmoothLine command..... 49
 - SnapPourPoint command..... 115
 - SnapPour. *See* SnapPourPoint command
 - Solve command..... 98
 - SpatialAutocorrelation command..... 71
 - Spatial Analyst toolbox..... 101
 - Spatial grid index
 - calculating..... 42
 - Spatial reference
 - creating..... 53
 - Spatial Statistics toolbox..... 71
 - Spline command..... 83, 117
 - SplitLine command..... 46
 - Split command..... 5, 21
 - SQR. *See* Square command
 - SQRT. *See* SquareRoot command
 - SquareRoot command..... 126
 - Square command..... 126
 - StandardDistance command..... 74
 - StandardizeAddresses command..... 68
 - Standardizing
 - addresses..... 68
 - Standard deviational ellipse..... 73
 - Statistics
 - calculating for a block..... 140
 - calculating for raster..... 56
 - calculating summary..... 9
 - frequency..... 8
 - of a cell..... 119
 - toolset..... 8
 - Statistics command..... 9
 - StreamLink command..... 115
 - StreamOrder command..... 115
 - StreamShape. *See* StreamToFeature command
 - StreamToFeature command..... 115
 - Subtract. *See* Minus command
 - Subtypes toolset..... 60
 - Sum. *See* CellStatistics command
 - Summary Statistics. *See* Statistics: command
 - Summing
 - two raster datasets..... 125
 - Surface
 - aspect..... 87, 144
 - slope..... 89, 145
 - toolset..... 144
 - SurfaceLength command..... 81
 - SurfaceSpot command..... 81
 - SurfaceVolume command..... 82
 - SymDiff command..... 6
 - Symmetrical difference..... 6
 - Syntax example..... 2
- ## T
- Tables
 - calculating frequency..... 8
 - converting or copying to tables..... 17
 - converting to dBASE..... 15, 17
 - copying..... 47
 - creating..... 61
 - creating from domains..... 41
 - creating pivot tables..... 62
 - deleting..... 47, 48
 - importing to geodatabases..... 17
 - making table views..... 52
 - renaming..... 48
 - selecting..... 5
 - TableSelect command..... 5
 - Tables toolset..... 35
 - TableToDBASE command..... 15
 - TableToDomain command..... 41
 - TableToGeodatabase command..... 17
 - TableToRelationshipClass command..... 60
 - TableToTable command..... 17
 - Table toolset..... 61
 - TabulateArea command..... 146
 - TanH command..... 137
 - Tan command..... 137
 - Temporary layers
 - making. *See* Layers and Table Views toolset
 - Test command..... 133
 - Thiessen command..... 25
 - Thin command..... 111
 - TigerArc command..... 27
 - TigerTool command..... 26
 - TIGER conversion
 - advanced..... 26
 - basic..... 27
 - Times command..... 86, 127
 - TINArc. *See* FeatureClassToCoverage command

TinAspect command	90	Toolset (continued)	
TinContour command	91	Database	39
TinDomain command	79	Data Management	28
TinEdge command	79	Density	102
TINHull. <i>See</i> TINDomain command		described	1
TINLattice. <i>See</i> TINRaster command		Disconnected Editing	39
TINLines. <i>See</i> TINEdge command		Distance	103
TinNode command	79	Domains	40
TinPolygonTag command	80	Extract	5, 21
TinRaster command	80	Extraction	107
TINs		Feature	53
calculating aspect	90	Features	43
calculating contours	91	Feature Class	41
calculating slope	91	Fields	46
converting to raster datasets	80	From Coverage	25
creating	90	From Raster	13
editing	90	Functional Surface	81
extracting edges	79	General	47
extracting interpolation zones	79	Generalization	30, 49, 108
extracting nodes	79	Groundwater	111
extracting polygon tag information	80	Hydrology	113
extracting triangle polygons	80	Indexes	33, 50
TinSlope command	91	Interpolation	116
TINSpot. <i>See</i> SurfaceSpot command		Items	33
TinTriangle command	80	Joins	33, 51
TIN Creation toolset	90	Layers and Table Views	51
TIN Surface toolset	90	Local	119
Tolerance		Logical	129
command	36	Mapping Clusters	72
setting cluster tolerances	63	Map Algebra	121
Tolerances toolset	36	Masking	11
Tool		Math	122
described	1	Measuring Geographic Distributions	73
Toolbox		Multivariate	137
3D Analyst	79	Neighborhood	140
Analysis	5	Network Dataset	98
Cartography	11	Overlay	6, 22, 141
Conversion	13	Projections	34
Coverage	21	Projections and Transformations	53
Data Interoperability	93	Proximity	7, 23
Data Management	39	Raster	54, 56
described	1	Raster Creation	142
Geocoding	67	Raster Interpolation	82
Geostatistical Analyst	95	Raster Math	85
Linear Referencing	69	Raster Reclass	86
Network Analyst	97	Raster Surface	87
Spatial Analyst	101	Reclass	142
Spatial Statistics	71	Relationship Classes	60
Toolset		Subtypes	60
Aggregate	28	Surface	144
Analysis	21, 97	Table	61
Analyzing Patterns	71	Tables	35
Bitwise	127	TIN Creation	90
Composite Features	28	TIN Surface	90
Conditional	101	Tolerances	36
Conversion	25, 79	Topology	36, 62

Toolset (continued)

To CAD	14
To Coverage	15, 26
To dBASE	15
To Geodatabase	15
To Raster	17
To Shapefile	19
Trigonometric	134
Turn Feature Class	99
Utilities	75
Versions	63
Workspace	65
Workspace Management	37
Zonal	146
Topology	
adding feature classes	62
adding rules	62
creating	63
removing feature classes	63
removing rules	63
toolset	36, 62
validating	63
TopoToRasterByFile command	83, 117
TopoToRaster command	83, 117
To CAD toolset	14
To Coverage toolset	15, 26
To dBASE toolset	15
To Geodatabase toolset	15
To Raster toolset	17
To Shapefile toolset	19
Transforming	
raster datasets	54
route events	70
TransformRouteEvents command	70
Transform command	34
Trend command	84, 119
Trigonometric toolset	134
TurnTableToTurnFeatureClass command	99
Turn Feature Class toolset	99

U

Ungenerate command	26
Union command	7, 23
UnregisterAsVersioned command	64
Unregistering	
as versioned	64
UpdateAnnotation command	42
UpdateByAlternateIDFields command	99
UpdateByGeometry command	99
Update command	7, 23
Updating	
IDs	35
Utilities toolset	75

V

ValidateTopology command	63
Variety. <i>See</i> CellStatistics command	
Velocity. <i>See</i> DarcyVelocity command	
Versions	
altering	63
changing edit session	64
creating	64
deleting	64
reconciling	64
registering	64
toolset	63
unregistering	64
Viewshed command	89, 146
VisdeCode. <i>See</i> ObserverPoints command	
Visibility. <i>See</i> LineOfSight command; Viewshed command	
Volume. <i>See</i> SurfaceVolume command	
VPF	
creating tile topology	37
VPFExport command	26
VPFImport command	28
VPFTile command	37

W

Warp command	55
Watershed command	116
WeightedOverlay command	141
Workspace	
creating	65
toolset	65
Workspace Management toolset	37

X

XOr. *See* Exclusive or

Z

ZonalArea. <i>See</i> TabulateArea command	
ZonalCentroid. <i>See</i> ZonalGeometry command	
ZonalFill command	146
ZonalGeometryAsTable command	146
ZonalGeometry command	146. <i>See also</i> ZonalGeometryAsTable command
ZonalMajority. <i>See</i> ZonalStatistics command	
ZonalMax. <i>See</i> ZonalStatistics command	
ZonalMean. <i>See</i> ZonalStatistics command	
ZonalMedian. <i>See</i> ZonalStatistics command	
ZonalMin. <i>See</i> ZonalStatistics command	
ZonalMinority. <i>See</i> ZonalStatistics command	
ZonalPerimeter. <i>See</i> ZonalGeometry command	
ZonalRange. <i>See</i> ZonalStatistics command	
ZonalStatisticsAsTable command	147
ZonalStatistics command	147
ZonalStats. <i>See</i> ZonalStatisticsAsTable command	

ZonalStd.	<i>See</i> ZonalStatistics command
ZonalSum.	<i>See</i> ZonalStatistics command
ZonalThickness.	<i>See</i> ZonalGeometry command
ZonalVariety.	<i>See</i> ZonalStatistics command
Zonal toolset.....	146
ZRenderer command.....	76

Appendix A: Toolset licensing

The core toolboxes contain a mixture of tools available with specific license levels. These are shown in the tables below. The Coverage toolbox is only available with the ArcInfo license. All tools provided with extensions are available at any license level.

Analysis toolbox

Toolset/Tool	ArcView	ArcEditor	ArcInfo
Extract toolset			
Clip	✓	✓	✓
Select	✓	✓	✓
Split			✓
Table Select	✓	✓	✓
Overlay toolset			
Erase			✓
Identity			✓
Intersect	✓	✓	✓
Symmetrical Difference			✓
Union	✓	✓	✓
Update			✓
Proximity toolset			
Buffer	✓	✓	✓
Multiple Ring Buffer	✓	✓	✓
Near			✓
Point Distance			✓
Statistics toolset			
Frequency			✓
Summary Statistics	✓	✓	✓

Cartography toolbox

Tool	ArcView	ArcEditor	ArcInfo
Cul-de-Sac Masks			✓
Feature Outline Masks			✓
Intersecting Layers Masks			✓

Conversion Toolbox

Toolset/Tool	ArcView	ArcEditor	ArcInfo
From Raster			
Raster To ASCII	✓	✓	✓
Raster To Float	✓	✓	✓
Raster To Point	✓	✓	✓
Raster To Polygon	✓	✓	✓
Raster To Polyline	✓	✓	✓
To CAD			
Add CAD Fields			✓
Create CAD XData			✓
Export To CAD			✓
Set CAD Alias			✓
To Coverage			
Feature Class To Coverage			✓
To dBase			
Table To dBase (multiple)	✓	✓	✓
To Geodatabase			
Feature Class To Feature Class	✓	✓	✓
Feature Class To Geodatabase (multiple)	✓	✓	✓
Import CAD Annotation	✓	✓	✓
Import Coverage Annotation	✓	✓	✓
Import From CAD	✓	✓	✓
Raster To Geodatabase (multiple)	✓	✓	✓
Table To Geodatabase (multiple)	✓	✓	✓
Table To Table	✓	✓	✓
To Raster			
ASCII To Raster	✓	✓	✓
DEM To Raster	✓	✓	✓
Feature To Raster	✓	✓	✓
Float To Raster	✓	✓	✓
Raster To Other Format (multiple)	✓	✓	✓
To Shapefile			
Feature Class To Shapefile (multiple)	✓	✓	✓

Data Management toolbox

Toolset/Tool	ArcView	ArcEditor	ArcInfo
Database toolset			
Compact	✓	✓	✓
Compress		✓	✓
Disconnected Editing toolset			
Check In		✓	✓
Check In From Delta		✓	✓
Check Out		✓	✓
Export To Delta		✓	✓
Domains toolset			
Add Coded Value To Domain	✓	✓	✓
Assign Domain To Field	✓	✓	✓
Create Domain	✓	✓	✓
Delete Coded Value From Domain	✓	✓	✓
Delete Domain	✓	✓	✓
Domain To Table	✓	✓	✓
Remove Domain From Field	✓	✓	✓
Set Value For Range Domain	✓	✓	✓
Table To Domain	✓	✓	✓
Feature Class toolset			
Append Annotation Feature Classes	✓	✓	✓
Calculate Default Cluster Tolerance	✓	✓	✓
Calculate Default Spatial Grid Index	✓	✓	✓
Create Feature Class	✓	✓	✓
Integrate	✓	✓	✓
Update Annotation Feature Class	✓	✓	✓
Features toolset			
Add XY Coordinates	✓	✓	✓
Check Geometry	✓	✓	✓
Copy Features	✓	✓	✓
Delete Features	✓	✓	✓
Feature Envelope To Polygon			✓
Feature To Line			✓
Feature To Point			✓
Feature To Polygon			✓
Feature Vertices to Points			✓
Multipart To Singlepart	✓	✓	✓
Polygon To Line			✓
Repair Geometry	✓	✓	✓
Split Line At Vertices			✓

Fields toolset	ArcView	ArcEditor	ArcInfo
Add Field	✓	✓	✓
Assign Default To Field	✓	✓	✓
Calculate Field	✓	✓	✓
Delete Field	✓	✓	✓
General toolset			
Append	✓	✓	✓
Copy	✓	✓	✓
Delete	✓	✓	✓
Merge	✓	✓	✓
Rename	✓	✓	✓
Select Data	✓	✓	✓
Generalization toolset			
Dissolve	✓	✓	✓
Eliminate			✓
Simplify Line		✓	✓
Smooth Line		✓	✓
Indexes toolset			
Add Attribute Index	✓	✓	✓
Add Spatial Index	✓	✓	✓
Remove Attribute Index	✓	✓	✓
Remove Spatial Index	✓	✓	✓
Joins toolset			
Add Join	✓	✓	✓
Remove Join	✓	✓	✓
Layers and Table Views toolset			
Make Feature Layer	✓	✓	✓
Make Query Layer	✓	✓	✓
Make Raster Catalog Layer	✓	✓	✓
Make Raster Layer	✓	✓	✓
Make Table View	✓	✓	✓
Make XY Event Layer	✓	✓	✓
Save To Layer File	✓	✓	✓
Select Layer By Attribute	✓	✓	✓
Select Layer By Location	✓	✓	✓
Projections and Transformations toolset			
Define Projection	✓	✓	✓
Feature toolset			
Batch Project	✓	✓	✓
Create Spatial Reference	✓	✓	✓
Project	✓	✓	✓

Raster toolset	ArcView	ArcEditor	ArcInfo
Flip	✓	✓	✓
Mirror	✓	✓	✓
Project Raster	✓	✓	✓
Rescale	✓	✓	✓
Rotate	✓	✓	✓
Shift	✓	✓	✓
Warp	✓	✓	✓
Raster toolset			
Batch Build Pyramids	✓	✓	✓
Batch Calculate Statistics	✓	✓	✓
Build Pyramids	✓	✓	✓
Calculate Statistics	✓	✓	✓
Clip	✓	✓	✓
Composite Bands	✓	✓	✓
Copy Raster	✓	✓	✓
Copy Raster Catalog Items	✓	✓	✓
Create Raster Catalog	✓	✓	✓
Create Raster Dataset	✓	✓	✓
Delete Raster Catalog Items	✓	✓	✓
Mosaic	✓	✓	✓
Mosaic To New Raster	✓	✓	✓
Resample	✓	✓	✓
Relationship Classes toolset			
Create Relationship Class		✓	✓
Table To Relationship Class		✓	✓
Subtypes toolset			
Add Subtype	✓	✓	✓
Remove Subtype	✓	✓	✓
Set Default Subtype	✓	✓	✓
Set Subtype Field	✓	✓	✓
Table toolset			
Analyze		✓	✓
Change Privileges		✓	✓
Copy Rows	✓	✓	✓
Create Table	✓	✓	✓
Delete Rows	✓	✓	✓
Get Count	✓	✓	✓
Pivot Table			✓

Topology toolset	ArcView	ArcEditor	ArcInfo
Add Feature Class To Topology		✓	✓
Add Rule To Topology		✓	✓
Create Topology		✓	✓
Remove Feature Class From Topology		✓	✓
Remove Rule From Topology		✓	✓
Set Cluster Tolerance		✓	✓
Validate Topology		✓	✓
Versions toolset			
Alter Version		✓	✓
Create Version		✓	✓
Delete Version		✓	✓
Post Version		✓	✓
Reconcile Version		✓	✓
Register As Versioned		✓	✓
Unregister As Versioned		✓	✓
Workspace toolset			
Create ArcInfo Workspace			✓
Create Feature Dataset	✓	✓	✓
Create Folder	✓	✓	✓
Create Personal Geodatabase	✓	✓	✓

Geocoding toolbox

Tool	ArcView	ArcEditor	ArcInfo
Automate Geocoding Indexes		✓	✓
Create Address Locator	✓	✓	✓
Deautomate Geocoding Indexes		✓	✓
Delete Address Locator	✓	✓	✓
Geocode Addresses	✓	✓	✓
Rebuild Geocoding Indexes	✓	✓	✓
Standardize Addresses	✓	✓	✓

Linear Referencing toolbox

Tool	ArcView	ArcEditor	ArcInfo
Calibrate Routes	✓	✓	✓
Create Routes	✓	✓	✓
Dissolve Route Events	✓	✓	✓
Locate Features Along Routes	✓	✓	✓
Make Route Event Layer	✓	✓	✓
Overlay Route Events	✓	✓	✓
Transform Route Events	✓	✓	✓

Spatial Statistics toolbox

Toolset/Tool	ArcView	ArcEditor	ArcInfo
Analyzing Patterns toolset			
Average Nearest Neighbor	✓	✓	✓
High/Low Clustering (Getis-Ord General G)	✓	✓	✓
Spatial Autocorrelation (Moran's I)	✓	✓	✓
Mapping Clusters toolset			
Cluster And Outlier Analysis (Anselin Local Moran's I)	✓	✓	✓
Cluster/Outlier Analysis With Rendering	✓	✓	✓
Hot Spot Analysis (Getis-Ord Gi*)	✓	✓	✓
Hot Spot Analysis With Rendering	✓	✓	✓
Measuring Geographic Distributions toolset			
Central Feature	✓	✓	✓
Directional Distribution (Standard Deviation Ellipse)	✓	✓	✓
Linear Directional Mean	✓	✓	✓
Mean Center	✓	✓	✓
Standard Distance	✓	✓	
Utilities toolset			
Calculate Areas	✓	✓	✓
Collect Events	✓	✓	✓
Collect Events With Rendering	✓	✓	✓
Count Rendering	✓	✓	✓
Export Feature Attribute To ASCII	✓	✓	✓
Z Score Rendering	✓	✓	✓

