

ArcGIS® 9

Geoprocessing Commands Quick Reference Guide



Copyright © 2004-2008 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, Catherine Jones, Corey Tucker

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	17
Coverage toolbox	27
Data Management toolbox	45
Geocoding toolbox	85
Linear Referencing toolbox	87
Mobile toolbox	89
Multidimension toolbox	91
Server toolbox	93
Spatial Statistics toolbox	97
3D Analyst toolbox	109
Data Interoperability toolbox	129
Geostatistical Analyst toolbox	131
Network Analyst toolbox	133
Schematics toolbox	137
Spatial Analyst toolbox	139
Tracking Analyst toolbox	189
Index	191
Appendix A: Tool licensing	A-1





Introduction

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

All commands (otherwise known as tools) 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.

There are four different kinds of tools, and they differ only in how they are created and added to a toolbox. However, regardless of their type, all tools can be executed from their dialog or from the command line and can be used in models and scripts.

-  System tool—these tools are installed and registered on your system. Usually, these tools are installed and registered when you install ArcGIS, although third-party developers can also create and register system tools. System tools are sometimes called function tools by developers.
-  Model tool—these tools are created by you with ModelBuilder. Some of the tools in the system toolboxes are model tools.
-  Script tool—these tools are created by you with a scripting language editor (typically an enhanced text editor). Some of the tools in the system toolboxes are script tools.
-  Custom tool—custom tools are built by system developers and have their own unique user interface for creating the tool. The ArcGIS Data Interoperability extension contains custom tools.

This guide describes the following toolboxes:

Analysis toolbox	Multidimension toolbox
Cartography toolbox	Mobile toolbox
Conversion toolbox	Network Analyst toolbox
Coverage toolbox	Schematics toolbox
Data Interoperability toolbox	Server toolbox
Data Management toolbox	Spatial Analyst toolbox
Geocoding toolbox	Spatial Statistics toolbox
Geostatistical Analyst toolbox	Tracking Analyst toolbox
Linear Referencing 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 with which they are associated. 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. Examples of alias usages are 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	Multidimension toolbox	_md
Cartography toolbox	_cartography	Mobile	_mobile
Conversion toolbox	_conversion	Network Analyst toolbox	_na
Coverage toolbox	_arc	Schematics	_schematics
Data Interoperability toolbox	_di	Server toolbox	_server
Data Management toolbox	_management	Spatial Analyst toolbox	_sa
Geocoding toolbox	_geocoding	Spatial Statistics toolbox	_stat
Geostatistical Analyst toolbox	_ga	Tracking Analyst toolbox	_ta
Linear Referencing toolbox	_l	3D Analyst toolbox	_3d

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 that follow 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

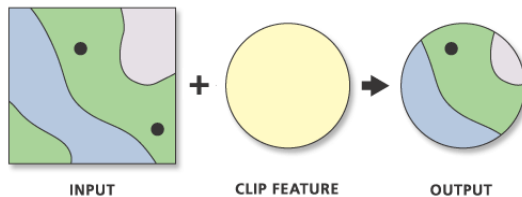
A suite of geoprocessing tools used to solve spatial or statistical problems.

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.

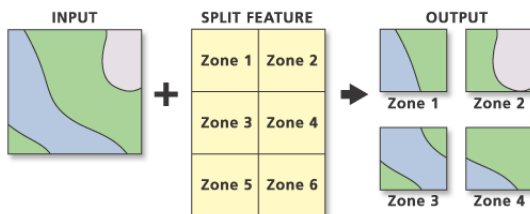
- ❖ **Select:** Extracts selected features from an input feature class or layer and stores them in the output feature class.

`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.

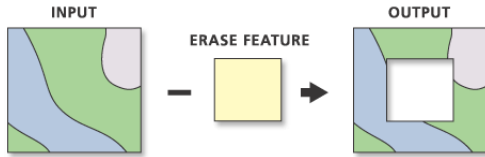


Overlay toolset

Contains tools for topological integration of features based on symmetry.

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

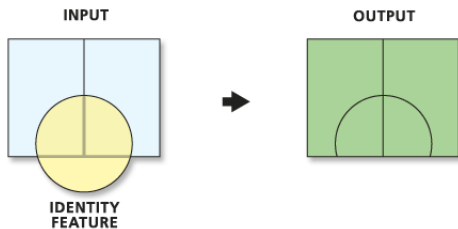
`Erase <in_features> <erase_features> <out_feature_class> {cluster_tolerance}`



- 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.

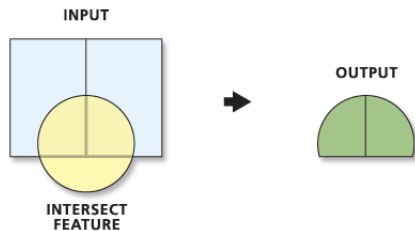
`Identity <in_features> <identity_features> <out_feature_class> {ALL | NO_FID | ONLY_FID} {cluster_tolerance} {NO_RELATIONSHIPS | KEEP_RELATIONSHIPS}`



- 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}`

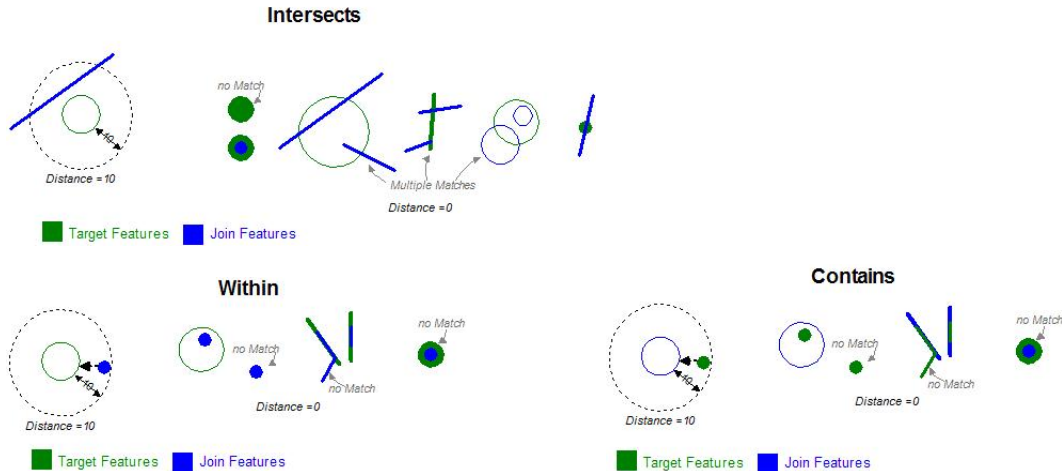


- The input features must be point, multipoint, line, or polygon. The inputs cannot be annotation features, dimension features, or network features.
- If the inputs have different geometry types (that is, line on poly, point on line, and so on), the output feature class geometry type will default to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

❖ **Spatial Join:** Creates a type of table join in which fields from one layer's attribute table are appended to another layer's attribute table based on the relative locations of the features in the two layers.

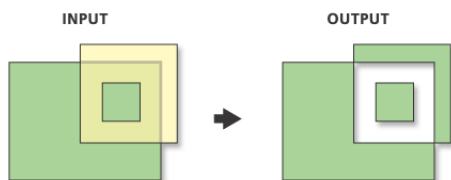
`SpatialJoin <target_features> <join_features> <out_feature_class> {JOIN_ONE_TO_ONE | JOIN_ONE_TO_MANY} {KEEP_ALL | KEEP_COMMON} {field_mapping} {INTERSECTS | IS_WITHIN | CONTAINS | CLOSEST} {search_radius} {distance_field_name}`

- A count field is added to the output for all joins where the join operation is ONE_TO_ONE. For nearest joins, a distance field is also added to the output.
- The distance field name is the name of the field in the output feature class that defines the distance between the target feature and the join feature. This field is only added with the closest match option is used.
- Visually the map options look as follows:



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

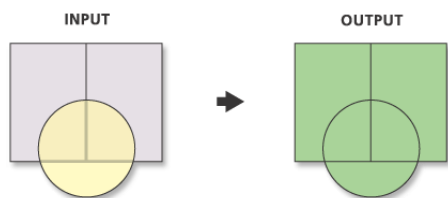
`SymDiff <in_features> <update_features> <out_feature_class> {ALL | NO_FID | ONLY_FID}`
`{cluster_tolerance}`



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

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

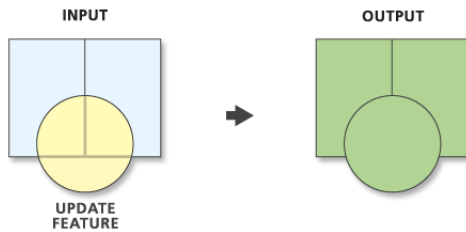
`Union <features {Ranks}; features {Ranks}...> <out_feature_class> {ALL | NO_FID | ONLY_FID}`
`{cluster_tolerance} {GAPS | NO_GAPS}`



- All input feature classes and feature layers must have polygon geometry.
- With ArcView and Editor licenses, the number of input feature classes or layers is limited to two.

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.

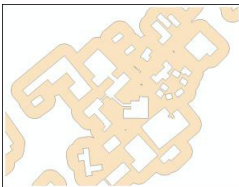


Proximity toolset

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.

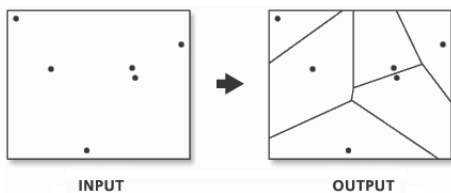
`Buffer <in_features> <out_feature_class> <buffer_distance_or_field> {FULL | LEFT | RIGHT | OUTSIDE_ONLY} {ROUND | FLAT} {NONE | ALL | LIST} {dissolve_field;dissolve_field...}`



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

Create Thiessen Polygons: Converts input points to an output feature class of Thiessen proximal polygons.

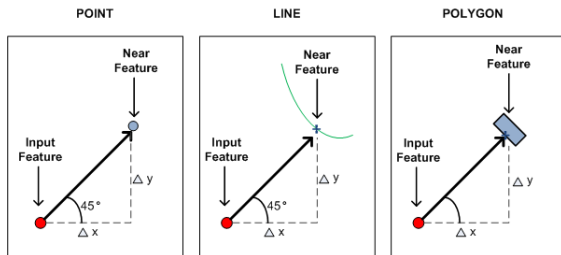
`CreateThiessenPolygons <in_features> <out_feature_class> <ONLY_FID | ALL>`



- Thiessen polygons have the unique property that each polygon contains only one input point, and any location within a polygon is closer to its associated point than to the point of any other polygon.
- Thiessen polygons can be used to apportion a point coverage into polygons known as Thiessen or Voronoi polygons.

Generate Near Table: Determines the distance from each point in the input to the nearest feature in the near feature class or layer, within the search radius, and outputs to a table.

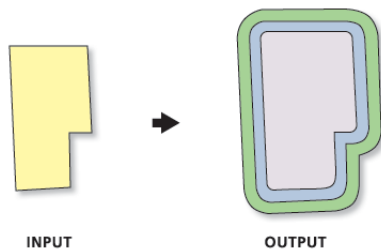
`GenerateNearTable <in_features> <near_features;near_features...> <out_table> {search_radius} {NO_LOCATION | LOCATION} {NO_ANGLE | ANGLE} {CLOSEST | ALL} {closest_count}`



- This behaves the same as NEAR but it will be required to create a new output table. The table will contain the INPUT_FID, NEAR_FID and other attributes such as NEAR_XY, NEAR_ANGLE, NEAR_FC if necessary. The input features will not be updated.
- The output table can be joined back to the input feature class or a near feature class using the original FIDs.
- The <in_features> and <near_features> support any geometry type: point, polyline, polygon, or multipoint.
- The default option is to output only the closest features. However, you can choose the ALL option to create the table containing all the near features within the search radius.

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

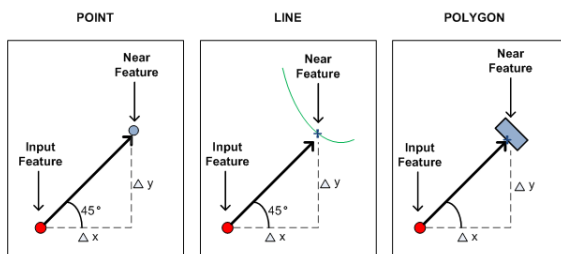
MultipleRingBuffer <input_features> <output_feature_class> <distances;distances...>
{DEFAULT | CENTIMETERS | DECIMALDEGREES | FEET | INCHES | KILOMETERS | METERS | MILES | MILLIMETERS | NAUTICALMILES | POINTS | YARDS} {field_name} {ALL | NONE} {FULL | OUTSIDE_ONLY}



- If the <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, within the search radius.

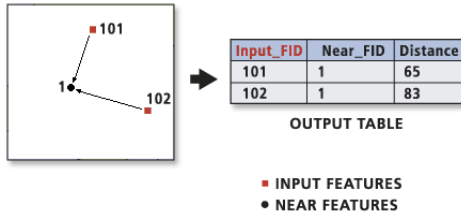
Near <in_features> <near_features;near_features...> {search_radius} {NO_LOCATION | LOCATION} {NO_ANGLE | ANGLE}



- 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.
- The <in_features> and <near_features> support any geometry type: point, polyline, polygon, or multipoint.

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 ID 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 field(s) and summary field(s).

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

`Statistics <in_table> <out_table> <statistics_field{Statistic Type};statistics_field{Statistic Type}...> {case_field;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 Add Field button, used to define the statistics fields, is used only in ModelBuilder™.
- Multiple case fields can be defined when calculating the statistics.

Cartography toolbox

Contains tools designed to produce data and support map production for specific maps in a way that meets a specific cartographic standard.

Graphic Quality toolset

Contains a tool that allows you to construct polygons that identify the location where two features share the same graphic space.

Detect Graphic Conflict: Detects graphic conflicts between feature representations and stores the overlaps as polygons in the output feature class.

```
DetectGraphicConflict <in_features> <conflict_features> <out_feature_class> {conflict_distance} {line_connection_allowance}
```

- This tool works on feature representations only, not feature geometries. The input and conflict feature layers must contain representations; otherwise, the tool will not execute.
- The conflict calculation is based on a reference scale. If you access this tool from ArcMap™, the reference scale of the data frame containing the input layers will be used unless an explicit reference scale has been set in the ArcToolbox environment settings. If you access the tool from ArcCatalog™, a reference scale must be specified in the environment settings; otherwise, the tool will not execute.
- The output feature class stores polygons, each representing an area of graphic conflict between an input representation and a conflict representation.

Masking toolset

Contains tools to construct masking polygons to use with variable depth masking in ArcMap, which obscures some feature symbology to help make clearer and more legible maps.

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.
- Masks will be created if the margin distance is zero or negative. A margin size of zero 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 zero will be specified to produce the desired masking effect.



Representation Management toolset

Contains tools that manage feature class representations and representation overrides.

Add Representation: Adds a feature class representation to a feature class.

```
AddRepresentation <in_features> <representation_name> {rule_id_field_name} {override_
field_name} {STORE_CHANGE_AS_OVERRIDE | MODIFY_FEATURE_SHAPE} {import_rule_layer}
{ASSIGN | NO_ASSIGN}
```

- The input must be a geodatabase feature class.
- Specify an import rule layer to import representation rules from an existing layer file that symbolizes features with a feature class representation. All the representation rules of the import rule layer file will be copied into this feature class representation.
- If the import rule layer has the same source feature class as the input feature class, you can check Assign Rule IDs (or use the ASSIGN option in a script or at the command line) to assign representation rules to features to match the RuleID assignments of the import rule layer.

Calculate Representation Rule: Applies existing representation rules to features in a feature class representation by calculating the RuleID field.

```
CalculateRepresentationRule <in_features> <representation> <representation_rule>
```

Drop Representation: Deletes a feature class representation from a feature class.

```
DropRepresentation <in_features> <representation>
```

- Once a feature class representation is deleted from a feature class, all representation rules and feature overrides associated with that representation are deleted.

Remove Override: Removes geometry overrides and/or representation property overrides from a feature class representation.

```
RemoveOverride <in_features> <representation> {BOTH | GEOMETRY_OVERRIDE | REPRESENTATION_
PROPERTY_OVERRIDE}
```

- Once the override of a feature is removed, the standard representation rule will apply.

Select Feature By Override: Selects features within a feature class representation that have shape and/or representation property overrides.

`SelectFeatureByOverride <in_representations> {BOTH | GEOMETRY_OVERRIDE | REPRESENTATION_PROPERTY_OVERRIDE}`

Update Override: Moves feature representation overrides from the default override field to explicit fields as defined by the representation rules in a feature class representation.

`<in_features> <representation> {REPRESENTATION_PROPERTY_OVERRIDE | GEOMETRY_OVERRIDE | BOTH}`

- The fields to be updated must be added and associated with the correspondent override attributes prior to using this tool.



Symbolization Refinement toolset

Contains tools that allow you to enrich the symbology used by your representations, including the alignment and type of symbols, as well as create custom symbology for bridges and tunnels.

Align Marker To Stroke Or Fill: Aligns the representation marker symbols of a point feature class to the nearest stroke or fill representation symbols in a line or polygon feature class within a specified search distance.

`AlignMarkerToStrokeOrFill <in_point_features> <in_line_or_polygon_features> <search_distance> {PERPENDICULAR | PARALLEL}`

- Markers beyond the search distance are not affected.
- The changes will be stored as overrides.

Calculate Geodesic Angle: Calculates geodesic angles for the input features according to the defined coordinate system and assigns the angle values to the specified field in the feature class that contains the input features.

`CalculateGeodesicAngle <in_features> <angle_field>`

- The input features can be points, lines, or polygons. For a point feature, the point location will be used to calculate the geodesic angle. For a line or polygon feature, the center point (centroid) of geometry will be used.
- You can use the Add Field tool to add a numerical field to the feature class containing the input features to store the calculated values.
- The coordinate system used in the calculation is defined in the geoprocessing Environment Settings > Cartography Settings. If it is not defined, the tool will use the one from the map if you are in ArcMap, or it will not execute if you are in ArcCatalog.
- The stored angles are in decimal degrees.

Calculate Line Caps: Calculates the cap type (ending style) for double-line symbols in the input representations.

`CalculateLineCaps <in_representations> {BUTT | SQUARE} {CASED_LINE_DANGLE | TRUE_DANGLE}`

- The line cap type defines how the ends of line segments are drawn using a double-line symbol. By default, a round line cap will be used. You can use this tool to change the cap type to BUTT or SQUARE.
- The calculated cap types will be stored as representation property overrides. To change back to round caps, you can just remove the representation property overrides.

- The dangle options are very specific. The TRUE_DANGLE means the end of a linear feature, for example, the dead-end of a road. When this option is used, the tool will calculate line caps for true dangles only. The CASED_LINE_DANGLE is where a linear feature still continues, but the representation changes from a double-line or cased-line symbol to a single-line symbol. When this option is used, the tool will calculate line caps for both true dangles and cased-line dangles.

Calculate Polygon Main Angle: Calculates the main angles of the input polygon features and assigns the angle values to the specified field in the feature class that contains the polygon features.

`CalculatePolygonMainAngle <in_features> <angle_field>`

- You can use the Add Field tool to add a numerical field in the feature class containing the input features to store the calculated values.
- The longest side of a polygon is considered the main axis of the polygon. The angle of the main axis will be the main angle of the polygon.
- The stored angles are in decimal degrees.

Create Overpass: Generates mask polygons at the intersections of stroke representations to symbolize one set of strokes passing above the other.

`CreateOverpass <in_above_features> <in_below_features> <margin_along> <margin_across> <out_overpass_feature_class> <out_mask_relationship_class> {where_clause} {out_decoration_feature_class} {ANGLED | PARALLEL | NONE} {wing_tick_length}`

- Requires the indication of features participating in the creation of an overpass.
- Feature classes without representations are not supported by this tool.
- When above and below representations are the same, a SQL expression is encouraged for further refinement of feature selection.
- Overpass masks are created based on the user-indicated margin sizes.
- Existing overpass feature classes and existing mask relationship classes will not be overwritten if the same name is specified.

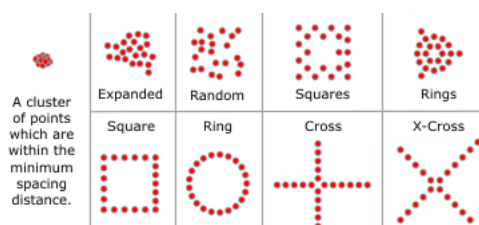
Create Underpass: Generates mask polygons at the intersections of stroke representations to symbolize one set of strokes passing under the other.

`CreateUnderpass <in_above_features> <in_below_features> <margin_along> <margin_across> <out_underpass_feature_class> <out_mask_relationship_class> {where_clause} {out_decoration_feature_class} {ANGLED | PARALLEL | NONE} {wing_tick_length}`

- Requires the indication of features participating in the creation of an underpass.
- Feature classes without representations are not supported by this tool.
- When above and below representations are the same, a SQL expression is encouraged for further refinement of feature selection.
- Underpass masks are created based on the user-indicated margin sizes.

Disperse Markers: Finds representation markers that are overlapping or too close to one another and spreads them apart based on a minimum spacing and dispersal pattern.

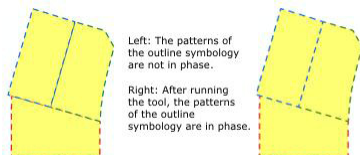
`DisperseMarkers <in_point_features> <minimum_spacing> {EXPANDED | RANDOM | SQUARES | RINGS | SQUARE | RING | CROSS | X_CROSS}`



- This tool is typically used when analyzing and symbolizing geocoded event data, where several events may happen at the same location or close locations, and become superimposed or overlapping when symbolized.
- The tool searches for coincident or nearly coincident (clustered) markers based on a tolerance derived from the specified Minimum Spacing and a reference scale; therefore, a reference scale must be set in order for the tool to execute.

Set Representation Control Points at Intersect: Finds a vertex along a line or polygon boundary that is shared by one or more features and sets the vertex as a representation control point.

`SetRepresentationControlPointAtIntersect <in_line_or_polygon_features> {in_features}`



- The result is stored as a geometry override. These control points can then be used to control the phasing of patterned representation symbology like dashed lines.
- This tool is appropriate for boundary symbology and any other features with a shared edge, since matching the phase of boundary symbology patterns on adjacent features otherwise involves manual manipulation of features.
- The primary input must be a line or polygon feature layer that is symbolized with a feature class representation.
- The secondary input can be a point, line, or polygon feature class; it does not need to have a feature class representation.
- The secondary input features only provide geometry for comparison to the primary input features.
- The secondary input receives representation control points only if it contains a feature class representation. This allows features from both inputs to receive representation control points simultaneously.
- If the secondary input is not specified, the tool operates on the primary input only, allowing self-intersecting features to be processed.

Set Representation Control Point By Angle: Finds a vertex along a line or polygon boundary where the inner angle is equal to or less than the specified Maximum Angle, sets the vertex as a representation control point, and stores the result as a geometry override.

`SetRepresentationControlPointByAngle <in_features> <maximum_angle>`

- An inner angle is the angle between the two line segments, measured less than 180 degrees, at a vertex. The smaller the inner angle is, the sharper turn it indicates.

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 into 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 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.



From WFS toolset

- ◆ **WFS to Feature Class:** Imports a feature type from a web feature service (WFS) to a feature class in a geodatabase.

`WFSToFeatureclass <input_WFS_server> <WFS_feature_type> <out_path> <out_name>`

- Upon setting the URL for the WFS server, all feature types published from the server are listed. Examples can include WFS features types for schools, roads and parcels. One of these is then selected and an output location and feature class name are specified. The output location can be the root level of a geodatabase or a feature dataset within a geodatabase.
- By default all features from the WFS source are added to the feature class. The extent environment setting can be used to limit the features to just those that intersect a user defined extent. You can also specify an output config keyword and output spatial grids using the geodatabase settings section of the environment settings.



Metadata toolset

Contains the tools to validate the metadata content according to a specific metadata standard or to export the metadata content to stand-alone metadata files that can be used with other metadata software.

- ❖ **ESRI Metadata Translator:** Uses ESRI's metadata translation engine to export metadata content from ArcCatalog to a stand-alone file. This tool requires the Microsoft .NET Framework 2.0. These tools are installed with the .NET Support install option.

`ESRITranslator <source> <translator> {output} {logfile}`

- ❖ **Metadata Importer:** Copies an XML metadata document as-is from the source item to the target item. This tool requires the Microsoft .NET Framework 2.0. These tools are installed with the .NET Support install option.

`MDImporter <source> <target>`

- ❖ **Metadata Publisher:** Publishes an XML metadata document to a metadata catalog such as an ArcIMS Metadata Service. This tool requires the Microsoft .NET Framework 2.0. These tools are installed with the .NET Support install option.

`MDPublisher <source> <publisher> {url} {service} {user} {password}`

- If you connect to ArcIMS in ArcCatalog with the same user name and password provided with this tool, the Metadata Service's icon will show a hand holding a pencil if you have permission to publish documents to that service.

- ❖ **USGS MP Metadata Translator :** Uses metadata parser to export or validate metadata content created using the FGDC metadata editor. This tool requires the Microsoft .NET Framework 2.0. These tools are installed with the .NET Support install option.

`USGSMPTTranslator <source> {config} {XML | NONE | HTML | TEXT | FAQ | SGML | DIF} {output} {errors}`

- Documentation for the metadata parser utility can be found on the USGS Web site.
- An metadata parser configuration file can be used to specify an extension file that will recognize metadata elements that are defined in FGDC CSDGM profiles. Extension and configuration files that will recognize ESRI-defined metadata elements and ESRI-ISO metadata elements can be found at ESRI's metadata Web page.
- When metadata is edited using ArcCatalog, the XML elements will not be in the correct order as specified by the FGDC CSDGM rules. As a result, metadata parser will record warnings in the error file.

- ❖ **XSLT Translator:** Uses an XML parser to transform an XML metadata document using an XSLT style sheet and export the result to a stand-alone file. This tool requires the Microsoft .NET Framework 2.0. These tools are installed with the .NET Support install option.

`XSLTranslator <source> <xslt> <output>`



To CAD toolset

Contains tools to convert geodatabase features to native CAD formats.

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>`

- All input feature classes and/or feature layers are valid inputs to this tool.
- XData is only read by AutoCAD.
- The XDList field that is created by this function is read by the Export to CAD tool when exported to AutoCAD.

Export to CAD: Creates one or more CAD drawings based on the values contained in one or more input feature classes or feature layers and supporting tables.

`ExportCAD <in_features;in_features...> <DWG_R2000 | DGN_V8 | DWG_R14 | DXF_R14 | DXF_R2000 | DWG_R2004 | DXF_R2004 | DWG_R2005 | DXF_R2005> <Output_File> {IGNORE_FILENAMES_IN_TABLES | USE_FILENAMES_IN_TABLES} {OVERWRITE_EXISTING_FILES | APPEND_TO_EXISTING_FILES} {Seed_File}`

- All input feature classes and/or feature layers and shapefiles are valid inputs to this tool.
- This tool 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.

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 Assign CAD Alias tool is an efficient way to have the Export to CAD tool recognize those values as CAD properties.
- Shapefiles are not 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 a tool 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 a tool 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.
- Copy Rows and Table To Table 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 into a geodatabase or to a shapefile.

```
FeatureClassToFeatureclass <in_features> <out_path> <out_name> {where_clause}
{field_mapping} {configuration_keyword}
```

- 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. 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 into a geodatabase annotation feature class.

```
ImportCADAnnotation <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 from your annotation.
- 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...> <output_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 SDE 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 (for example, gondor_1).

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

`TableToTable <in_rows> <out_path> <out_name> {where_clause} {field_mapping}
{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 KML toolset

Contains tools used to convert to Keyhole Markup Language (KML) files.

❖ **Layer to KML:** Converts an in-memory or file-based feature or raster layer into a Keyhole Markup Language (KML) file containing a translation of ESRI geometries and symbology into KML. This file is compressed using zip compression and will have a “.KMZ” extension and can be read by any KML client including ArcGIS Explorer, ArcGlobe, and Google Earth.

`LayerToKML <layer> <out_kmz_file> <layer_output_scale> {No_COMPOSITE | COMPOSITE}
{boundary_box_extent} {image_size} {dpi_of_client}`

- The output KMZ file cannot already exist.
- You can reduce the size of the output KMZ document if your layer has a scale-dependent renderer and you choose an appropriate “Layer Output Scale”.
- To output a single raster image draped over topography use the “Return single composite image” option.
- To output every layer as separate raster image use the “Convert Vector to Raster” option.

❖ **Map to KML:** Converts an in-memory Map or Map Document into a Keyhole Markup Language (KML) file containing a translation of ESRI geometries and symbology into KML. This file is compressed using zip compression and will have a “.KMZ” extension and can be read by any KML client including ArcGIS Explorer, ArcGlobe, and Google Earth.

`MapToKML <in_map_document> <data_frame> <out_kmz_file> <map_output_scale> {NO_COMPOSITE |
COMPOSITE} {VECTOR_TO_IMAGE | VECTOR_TO_VECTOR} {extent_to_export} {image_size} {dpi_ of_client}`

- The output KMZ file cannot already exist.
- You can reduce the size of the output KMZ document if your map has scale-dependent renderers and you choose an appropriate “Map Output Scale”.
- To output a single raster image draped over topography use the “Return single composite image” option.
- To output every layer as separate raster image use the “Convert Vector to Raster” option.



To Raster toolset

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

❖ **ASCII to Raster:** Converts an ASCII file representing raster data into 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 into 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 into 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 needs to 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.

❖ **Point to Raster:** Converts point features to a raster dataset.

PointToRaster <in_features> <value_field> <out_raster_dataset> {MOST_FREQUENT | SUM | MEAN | STANDARD_DEVIATION | MAXIMUM | MINIMUM | RANGE} {priority_field} {cellsize}

- Any shapefile, coverage, or geodatabase feature class containing point or multipoint features can be converted to a raster dataset.
- Multipoints are treated as individual points.
- The output cell size is the size of each pixel in the output raster dataset.
- Priority field is only used with the MOST_FREQUENT option.

❖ **Polygon to Raster:** Converts polygon features to a raster dataset.


PolygonToRaster <in_features> <value_field> <out_raster_dataset> {CELL_CENTER | MAXIMUM_AREA | MAXIMUM_COMBINED_AREA} {priority_field} {cellsize}

- Any shapefile, coverage, or geodatabase feature class containing polygon features can be converted to a raster dataset.
- If the input field contains floating-point values, the output raster will be floating point; if the input field contains integer values, the output raster will be integer; and if the input field contains string values, the output raster will contain an integer value field and a string field.
- The output cell size is the size of each pixel in the output raster dataset.

❖ **Polyline to Raster:** Converts polyline features to a raster dataset.

PolylineToRaster <in_features> <value_field> <out_raster_dataset> {MAXIMUM_LENGTH | MAXIMUM_COMBINED_LENGTH} {priority_field} {cellsize}

- Any shapefile, coverage, or geodatabase feature class containing polyline features can be converted to a raster dataset.
- If the input field contains floating-point values, the output raster will be floating point; if the input field contains integer values, the output raster will be integer; and if the input field contains string values, the output raster will contain an integer value field and a string field.
- The output cell size is the size of each pixel in the output raster dataset.

 ❖ **Raster to Other Format (multiple):** Converts one or more raster dataset formats supported by ArcGIS to a BMP, GIF, GRID, IMAGINE, JPEG, JPEG 2000, PNG, TIFF, or geodatabase raster dataset format.

`RasterToOtherFormat <input_rasters;input_rasters...> <output_workspace>`
{GRID | BMP | GIF | IMAGINE Image | JP2000 | JPEG | PNG | TIFF}

- The input raster datasets can be any valid raster dataset that ArcGIS can recognize.
- Allows you to batch convert raster datasets into another raster dataset format. This is useful if you receive raster datasets in one format, but you (or your client) prefer to use BMP, GIF, GRID, IMAGINE, JPEG, JPEG2000, PNG, or TIFF.

To Shapefile toolset

Contains a tool 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

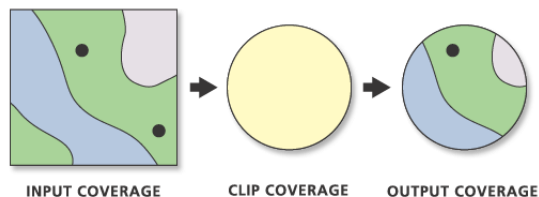
Contains tools and toolsets used for geospatial processing.

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.

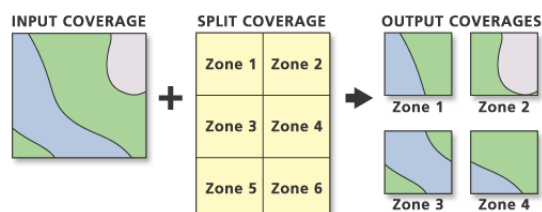
Select: Extracts features from the input coverage and stores them in an output coverage based on logical expressions or by applying the criteria contained in a selection file.

`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 coverage 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.

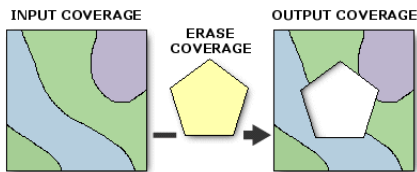


Overlay toolset

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.

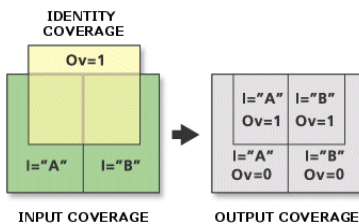
Erase <in_cover> <erase_cover> <out_cover> {POLY | LINE | POINT | NET | LINK | RAW}
{fuzzy_tolerance}



- 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.

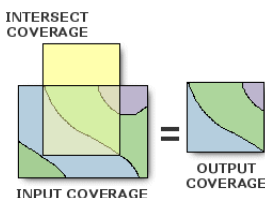
Identity <in_cover> <identity_cover> <out_cover> {POLY | LINE | POINT} {fuzzy_tolerance}
{JOIN | NO_JOIN}



- 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.

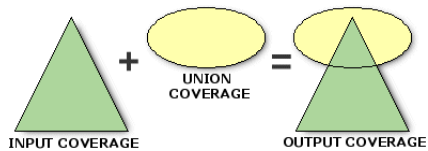
Intersect <in_cover> <intersect_cover> <out_cover> {POLY | LINE | POINT} {fuzzy_tolerance}
{JOIN | NO_JOIN}



- 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.

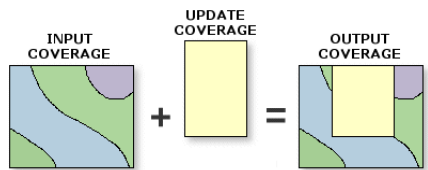
`Union <in_cover> <union_cover> <out_cover> {fuzzy_tolerance} {JOIN | NO_JOIN}`



- 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 only generated for the output coverage polygons 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.

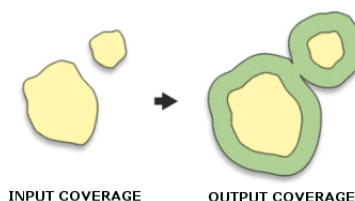


Proximity toolset

Contains tools used in geoprocessing analysis involving distance.

Buffer: Creates buffer polygons around specified input coverage features.

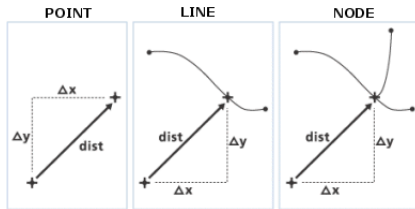
`Buffer <in_cover> <out_cover> {LINE | POLY | POINT | NODE} {buffer_item} {buffer_table} {buffer_distance} {fuzzy_tolerance} {ROUND | FLAT} {FULL | LEFT | RIGHT}`



- 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.

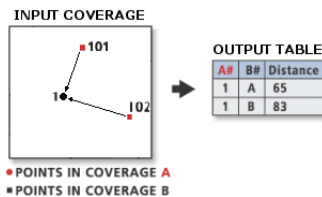
Near <in_cover> <near_cover> <out_cover> {LINE | POINT | NODE} {search_radius} {NO_LOCATION | LOCATION}



- 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.

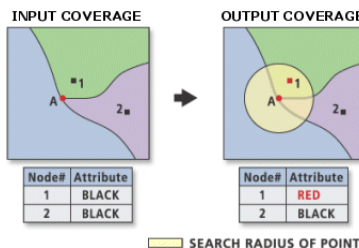
PointDistance <from_cover> <to_cover> <out_info_table> {search_radius}



- 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.

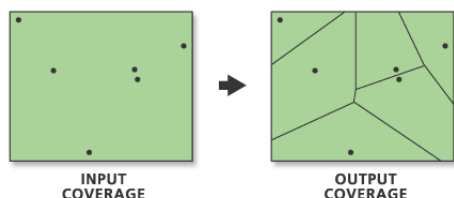
PointNode <point_cover> <node_cover> {search_radius}



- 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 (PAT) 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 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 (.e00).

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 is created in the process and 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 known as profiles. The TVP (designed specifically for planar vector data with topology), raster, and point profiles are the only profiles supported by Export to SDTS.
- 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 or it will not be exported.

Export to VPF: Converts a coverage into 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 pathname 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 into 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 a standard or Optional formatted Digital Line Graph 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 (.e00) into 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 ArcInfo 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 in addition to the data descriptive information.

- The Import From S57 tool creates either one or two ArcInfo 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 known as 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 U.S. Bureau of the Census TIGER data, 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 into an INFO table or converts a VPF coverage or tile into a coverage.

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

- Full VPF pathnames 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.

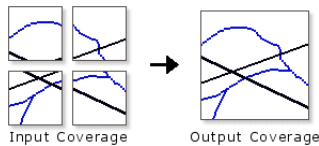


Aggregate toolset

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).

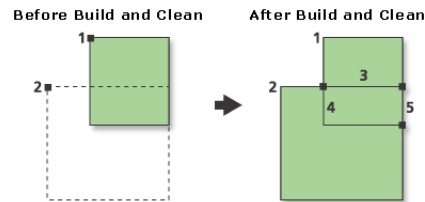


Composite Features toolset

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 is 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 polygon attribute table 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}`



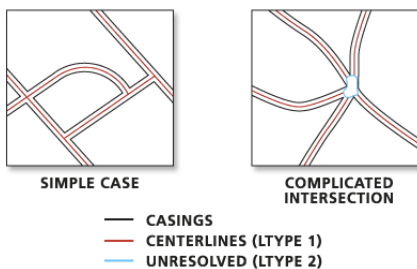
A) Nonorthogonal features

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.
- 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 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}`

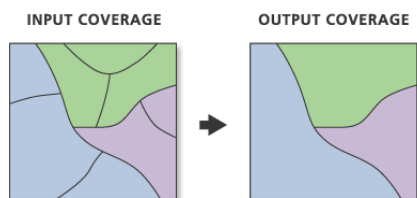


- In addition to the standard items, the output coverage AAT will contain the following five new items:
 - LTYPE—Contains a line type value of:
 - 1 centerlines
 - 2 unused lines and outlines of complicated intersections
 - 3 partition lines
 - LL#—Carries the left source arc record number.
 - RL#—Carries 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.

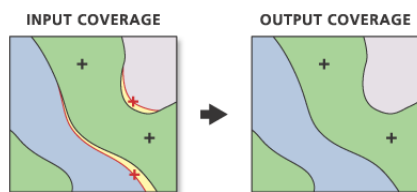
`Dissolve <in_cover> <out_cover> <dissolve_item> {POLY | LINE | NET | REGION.subclass}`



- 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}`

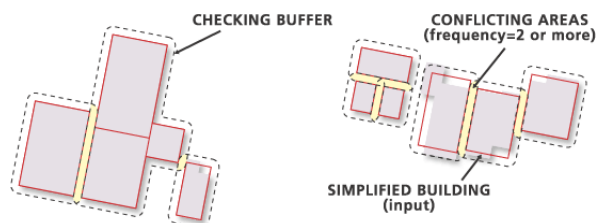


— ARCS TO BE ELIMINATED
+ LABEL POINTS TO BE ELIMINATED
■ SLIVER POLYGONS

- 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.

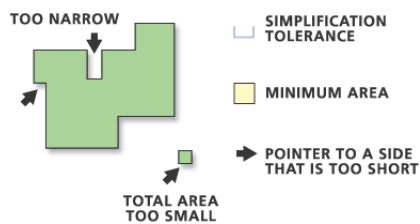
`FindConflicts <in_cover> <out_cover> <conflict_distance>`



- 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 buffers overlap indicates 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.

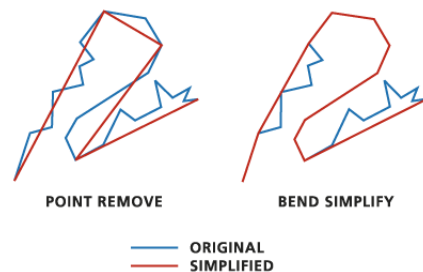
`SimplifyBuilding <in_cover> <out_cover> <simplification_tolerance> {minimum_area}
{selection_file} {NOT_CHECK | CHECK_CONFLICT}`



- 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 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.

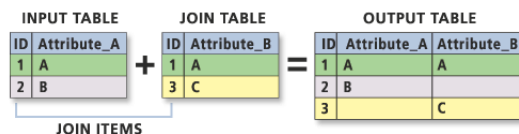


Joins toolset

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 it is 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.

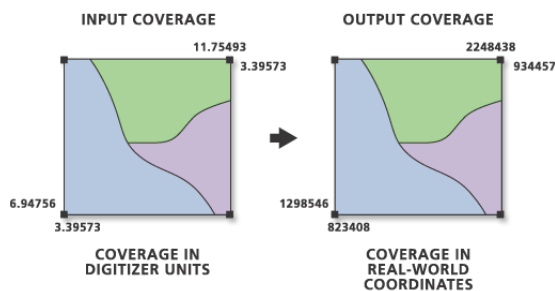
Project: Changes the coordinate system of a coverage including its datum or spheroid.

`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 coverages 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.

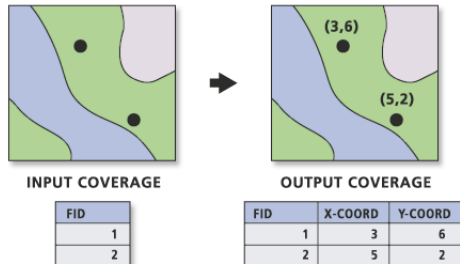


Tables toolset

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.

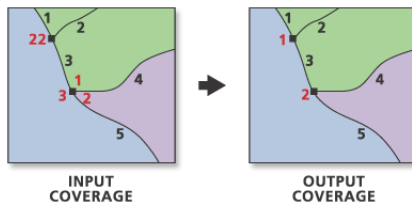
AddXY <in_cover> {POINT | NODE}



- 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 represent 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, 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.

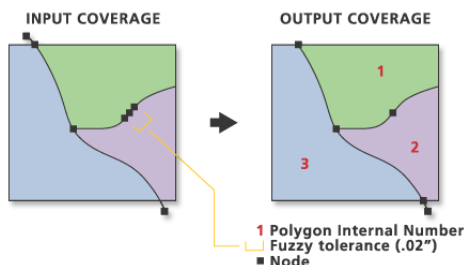
Build: Creates or updates feature attribute tables and polygon topology.

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 as 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.

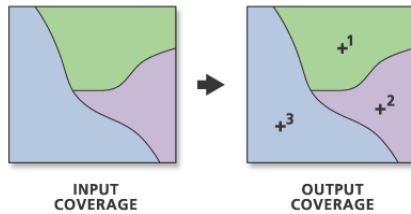
Clean <in_cover> {out_cover} {dangle_length} {fuzzy_tolerance} {POLY | LINE}



- 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 known as 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 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 may 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 ArcInfo 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, regardless of whether 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.

Data Comparison toolset

Contains tools to compare one dataset with another dataset and report any similarities and differences.

❖ **Feature Compare:** Compares two feature classes or layers and returns the comparison results.

```
FeatureCompare <in_base_features> <in_test_features> <sort_field;sort_field...> {ALL |  
  GEOMETRY_ONLY | ATTRIBUTES_ONLY | SCHEMA_ONLY | SPATIAL_REFERENCE_ONLY} {IGNORE_M |  
  IGNORE_Z | IGNORE_POINTID | IGNORE_EXTENSION_PROPERTIES | IGNORE_SUBTYPES | IGNORE_  
  RELATIONSHIPCLASSES | IGNORE_REPRESENTATIONCLASSES} {xy_tolerance} {m_tolerance}  
  {z_tolerance} {field {Tolerance};field {Tolerance}...} {omit_field;omit_field...}  
  {NO_CONTINUE_COMPARE | CONTINUE_COMPARE} {out_compare_file}
```

- The tool returns messages showing the comparison result. By default, the tool will stop executing after encountering the first miscompare. To report all differences, set the continue compare option to true.
- Multiple sort fields may be specified. The first field is sorted, then the second field, and so on, in ascending order. Sorting by a common field in both the input base features and the input test features ensures that you are comparing the same row from each input dataset.
- By default, the compare type is set to ALL.

❖ **File Compare:** Compares two files and returns the comparison results.

```
FileCompare <in_base_file> <in_test_file> {ASCII | BINARY} {NO_CONTINUE_COMPARE |  
  CONTINUE_COMPARE} {out_compare_file}
```

- The tool returns messages showing the comparison result. By default, the tool will stop executing after encountering the first miscompare. To report all differences, set the continue compare option to true.
- File Compare does support masking out of characters, words, and lines of text in an ASCII file.
- ASCII is the default file type. If entering binary files, change the file type to BINARY.
- The output compare file will contain all similarities and differences between the input base file and the input test file. This file is a comma-delimited text file that can be viewed and used as a table in ArcGIS.

❖ **Raster Compare:** Compares the properties of two rasters datasets or two raster catalogs and returns the comparison result.

```
RasterCompare <in_base_raster> <in_test_raster> {RASTER_DATASET | GDB_RASTER_DATASET |  
  GDB_RASTER_CATALOG} {ignore_option;ignore_option...} {NO_CONTINUE_COMPARE | CONTINUE_  
  COMPARE} {out_compare_file} {paramter{Tolerance}{Type};paramter{Tolerance}{Type}...} {f  
  ield{Tolerance};field{Tolerance}...} {omit_field;omit_field...}
```

- The tool returns messages showing the comparison result.

❖ **Table Compare:** Compares two tables and returns the comparison results.

```
TableCompare <in_base_table> <in_test_table> {sort_field;sort_field...} {ALL |  
  ATTRIBUTES_ONLY | SCHEMA_ONLY} {IGNORE_EXTENSION_PROPERTIES} {field {Tolerance};  
  field {Tolerance}...} {omit_field;omit_field...} {NO_CONTINUE_COMPARE | CONTINUE_COMPARE}  
  {out_compare_file}
```

- The tool returns messages showing the comparison result. By default, the tool will stop executing after encountering the first miscompare. To report all differences, set the continue compare option to true.
- Multiple sort fields may be specified. The first field is sorted, then the second field, and so on in ascending order. Sorting by a common field in both the input base table and the input test table ensures that you are comparing the same row from each input dataset.
- By default, the compare type is set to ALL.

- ❖ **TIN Compare:** Compares two triangulated irregular networks (TINs) and returns the comparison results.

`TinCompare <in_base_tin> <in_test_tin> {ALL | PROPERTIES_ONLY | SPATIAL_REFERENCE_ONLY} {NO_CONTINUE_COMPARE | CONTINUE_COMPARE} {out_compare_file}`

- The tool returns messages showing the comparison result. By default, the tool will stop executing after encountering the first miscompare. To report all differences, set the continue compare option to true.
- The output compare file will contain all similarities and differences between the input base features and the input test features. This file is a comma-delimited text file that can be viewed and used as a table in ArcGIS. For example, this table can be queried to obtain all the object ID values for all the rows that are different.



Database toolset

Contains tools that improve database performance.

- ❖ **Clear Workspace Cache:** Clears any ArcSDE workspaces from the ArcSDE workspace cache.

`ClearWorkspaceCache {in_data}`

- This tool can be used to help disconnect idle ArcSDE connections in a long running application.
- To clear the workspace cache correctly in a script, the call to `ClearWorkspaceCache()` should be the last call in your script, making sure to remove all references to any objects that may be pointing to the ArcSDE workspace before calling the tool.

- ❖ **Compact:** Reduces the size and optimizes the performance of a file or personal geodatabase.

`Compact <in_workspace>`

- You should compact your databases on a regular basis if you do a lot of data entry and deletion.
- Personal and file geodatabases are stored in binary files on your disk drive. As you add/remove/edit data in these files, they will become fragmented, resulting in a decrease in performance.
- It is recommended that you compact your personal geodatabase when its size increases to more than 250 MB.

- ❖ **Compress:** Compresses an ArcSDE geodatabase by removing states not referenced by a version and redundant rows.

`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.

- ❖ **Migrate Storage:** Changes how BLOB objects are stored within an ArcSDE geodatabase through the migration of BLOB objects using configuration keywords specified in the DBTUNE table.

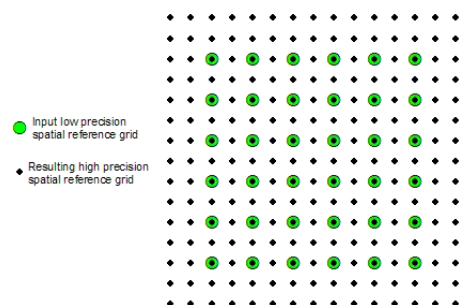
`MigrateStorage <in_datasets;in_datasets> <config_keyword>`

- It is recommended that you make a backup of the data before you migrate it.
- BLOBs are used to store geometries, attributes and raster information in geodatabase tables.

- Only certain migration paths are supported, for more information on the database specific migration paths supported consult the SDE documentation.

❖ **Upgrade Spatial Reference:** Upgrades a low precision dataset's spatial reference to high precision.

`UpgradeSpatialReference <input_dataset> {xy_resolution} {z_resolution} {m_resolution}`



- Valid input is stand-alone feature class, feature dataset, or raster catalog with low-resolution spatial reference and stored in an upgraded or current version of a personal or ArcSDE geodatabase.



Disconnected Editing toolset

Contains tools to allow remote or mobile users to transfer data from a central database to remote or mobile locations, enabling users to operate independently.

◆ **Check In:** Checks in changes made in a checkedout ArcSDE, file, or personal geodatabase to the master ArcSDE 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.
- Data cannot be checked in or checked out in ArcView.

◆ **Check In from Delta:** Checks in a geodatabase from a delta database or delta XML file. A delta file 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.
- When the check-in from the delta database or XML file succeeds, the checkout in the master geodatabase will be unregistered.
- Checking in changes from a delta database or XML file does not automatically unregister the checkout in the associated checkout geodatabase; this must be done manually.

◆ **Check Out:** Checks out datasets from an ArcSDE geodatabase to an ArcSDE, file, 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 is to a personal geodatabase that does not exist, one will be created.

- ◆ **Export to Delta:** Exports changes in a check-out geodatabase to a delta database or XML file. A delta file contains only the changes exported from a check-out 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.



Distributed Geodatabase toolset

Contains tools that allow organizations to disperse their data as necessary from central servers to regional or local offices that may be in a connected or disconnected environment.

Add Global IDs: Adds global IDs to a list of geodatabase classes, tables, and/or feature datasets.

`AddGlobalIDs <in_datasets;in_datasets...>`

- If the geodatabase is a personal or file geodatabase, then the datasets must be schema only.

Compare Replica Schema: Generates a report file XML comparing an input replica geodatabase and an XML schema file or geodatabase.

`CompareReplicaSchema <in_geodatabase> <in_source_file> <Output_replica_schema_changes_file>`

Create Replica: Creates a replica personal, file, or ArcSDE geodatabase from a specified list of feature classes, layers, datasets, and/or tables in an ArcSDE geodatabase.

`CreateReplica <in_data;in_data...> <TWO_WAY_REPLICA | ONE_WAY_REPLICA | CHECK_OUT> <out_geodatabase> <out_name> {FULL | SIMPLE} {CHILD_DATA_SENDER | PARENT_DATA_SENDER} {USE_DEFAULT_FILTERS | ADD_WITH_SCHEMA_ONLY | ALL_ROWS | DO_NOT_ADD} {DO_NOT_REUSE | REUSE} {GET_RELATED | DO_NOT_GET_RELATED}`

- All datasets must be from the same ArcSDE database.
- The spatial extent in the environment setting can be used to set the replica geometry. This can be set using another feature classes extent.
- The reuse schema option is only available for checkout replicas.

Create Replica Footprints: Creates a feature class that contains the geometries for all the replicas in a geodatabase.

`CreateReplicaFootPrints <in_geodatabase> <out_geodatabase> <output_feature_class_name>`

Create Replica From Server: Creates a replica using a specified list of feature classes, layers, datasets and/or tables from a remote geodatabase using a geodata service published on ArcGIS Server.

`CreateReplicaFromServer <in_geodataserver> <datasets;datasets...> <TWO_WAY_REPLICA | ONE_WAY_REPLICA | CHECK_OUT> <out_geodatabase> <out_name> {FULL | SIMPLE} {CHILD_DATA_SENDER | PARENT_DATA_SENDER} {USE_DEFAULT_FILTERS | ADD_WITH_SCHEMA_ONLY | ALL_ROWS | DO_NOT_ADD} {DO_NOT_REUSE | REUSE} {GET_RELATED | DO_NOT_GET_RELATED}`

Export Acknowledgement Message: Creates an output delta file to acknowledge the reception of a data change message from an input replica and replica geodatabase.

`ExportAcknowledgementMessage <in_geodatabase> <out_acknowledgement_file> <in_replica>`

- Checkout replicas are not listed.

Export Data Change Message: Creates an output delta file containing replica updates from an input replica and replica geodatabase.

`ExportDataChangeMessage <in_geodatabase> <out_data_changes_file> <in_replica> <DO_NOT_SWITCH | SWITCH> <UNACKNOWLEDGED | NO_UNACKNOWLEDGED> <NEW_CHANGES | NO_NEW_CHANGES>`

Export Replica Schema: Creates an output XML file with the schema of an input one- or two-way replica.

`ExportReplicaSchema <in_geodatabase> <output_replica_schema_file> <in_replica>`

Import Message: Imports changes from a delta file to a replica geodatabase.

`ImportMessage <in_geodatabase> <source_delta_file> {output_acknowledgement_file} {MANUAL | IN_FAVOR_OF_IMPORTED_CHANGES | IN_FAVOR_OF_DATABASE} {BY_OBJECT | BY_ATTRIBUTE} {DO_NOT_RECONCILE | RECONCILE}`

- After importing a data change message, you have the option to immediately export an acknowledgement message.

Import Replica Schema: Applies replica schema differences using an input replica geodatabase and XML schema file or geodatabase.

`ImportReplicaSchema <in_geodatabase> <in_source>`

Re-Export Unacknowledged Messages: Creates an output delta file containing unacknowledged replica updates from an input one- or two-way replica geodatabase.

`ReExportUnacknowledgedMessages <in_geodatabase> <output_delta_file> <in_replica> <ALL_UNACKNOWLEDGED | MOST_RECENT>`

- This command is not available for checkout replicas.

Synchronize Changes: Synchronizes updates between two replica geodatabases in a direction specified by the user.

`SynchronizeChanges <geodatabase_1> <in_replica> <geodatabase_2> <BOTH_DIRECTIONS | FROM_GEODATABASE2_TO_1 | FROM_GEODATABASE1_TO_2> <IN_FAVOR_OF_GDB1 | IN_FAVOR_OF_GDB2 | MANUAL> <BY_OBJECT | BY_ATTRIBUTE> <DO_NOT_RECONCILE | RECONCILE >`

- Once checked in, the changes (edits) will be reflected in the master geodatabase and viewable by all users.



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: 0.75—3/4", 2—2", 24—24", and 30—30".
- The coded value domain includes both the actual value that is stored in the database (for example, 1 for pavement) and a description of what the code value means (for example, pavement).

❖ **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...}`

- If no subtype is specified, the domain is only assigned to the specified field.
- 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, 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 of 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 designed to perform basic feature class management including creating, appending, integrating, and updating multiple feature classes.

❖ **Append Annotation Feature Classes:** Creates a new geodatabase annotation feature class by combining annotation from multiple input geodatabase annotation feature classes into a single feature class with annotation classes.

`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 into that feature class and the tool will project the annotation features into 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 or file geodatabase because it needs the existing index to maintain its integrity. When you create a new feature class in a personal or file 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, empty 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 a feature class created depends on the format of the pathname specified: a geodatabase feature class for ArcSDE and personal or file geodatabase pathnames 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.

❖ **Create Fishnet:** Creates a fishnet of rectangular cells.

`CreateFishnet <out_feature_class> <origin_coord> <y_axis_coord> <cell_width> <cell_
height> <number_rows> <number_columns> {corner_coord} {LABELS | NO_LABELS} {template}`

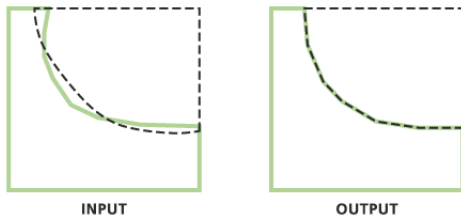
- If zero is specified for the cell size width and the cell size height, then specify the number of rows and columns, and the opposite corner of the fishnet (X, Y).
- The extent of the fishnet can be entered by specifying the coordinates or using a template dataset. When entering a template, the fishnet origin coordinate and y-axis coordinate are populated. You will still be required to enter the number of rows and columns.
- If zero is specified for the rows and columns, then specify the opposite corner of the fishnet (X, Y).
- If zero is entered for the cell size width or cell size height, then it will be automatically computed based on the number of rows and columns and the opposite corner of the fishnet.
- Labels are generated by default.
- To generate polygon features from the line features, you must use the Feature To Polygon tool.

❖ **Create Random Points:** Creates a user-specified number of random points in an extent, in the polygons of a feature class, or along the lines of a feature class.

```
CreateRandomPoints <out_path> <out_name> {constraining_feature_class} {constraining_extent} {number_of_points} {minimum_allowed_distance} {POINT | MULTIPOINT} {multipoint_size}
```

❖ **Integrate:** Compares features and makes any lines, points, or vertices within a certain 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.

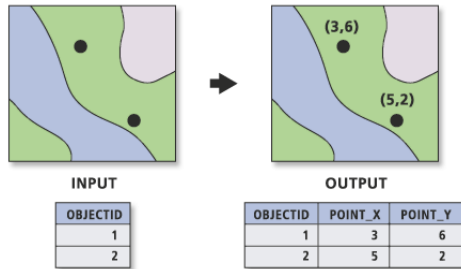


Features toolset

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 fields POINT_X and POINT_Y to the point input features and calculates their values.

AddXY <in_features>



- If the fields POINT_X and POINT_Y fields already exist, the values will be updated.
- 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 m-enabled.

- ❖ **Adjust 3D Z:** Allows the modification of all z-values in a z-enabled feature class.

Adjust3DZ <in_features> {NO_REVERSE | REVERSE} {adjust_value} {MILLIMETERS | CENTIMETERS | METERS | INCHES | FEET | YARDS | FATHOMS} {MILLIMETERS | CENTIMETERS | METERS | INCHES | FEET | YARDS | FATHOMS}

- Bathymetry data often has positive z-values. You may want to reverse the signs of all the data in the feature class to make it negative.
- Z-enabled data could be referenced to a vertical datum that is not appropriate for your geoprocessing needs. This tool could apply a bulk-shift of all the z-values in the feature class to adjust the data either up or down vertically.

- ❖ **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 or object ID 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 multipoint features, only the null geometry and empty part problems apply.
- For point features, only the null geometry problem applies.

- ❖ **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}

- The input features (geometry and attributes) will be copied to the output feature class.
- This tool can be used for data conversion as it can read many feature formats (any you can add to map) and write these out to shapefile or geodatabase (file, personal, or ArcSDE).
- If the output feature class already exists, it will be overwritten.

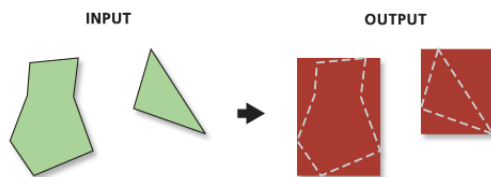
❖ **Delete Features:** Deletes features from the input feature class or layer.

`DeleteFeatures <in_features>`

- Both the geometry and the attributes will be deleted from the affected features.
- ArcSDE, file, 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 feature class.

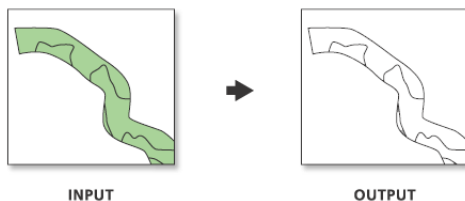
`FeatureEnvelopeToPolygon <in_features> <out_feature_class> {SINGLEPART | MULTIPART}`



- Valid inputs are line, polygon, and annotation feature classes.
- When the envelope of a feature is an invalid polygon (height or width is zero), the polygon will not 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 a new output line feature class from input polygon or line features.

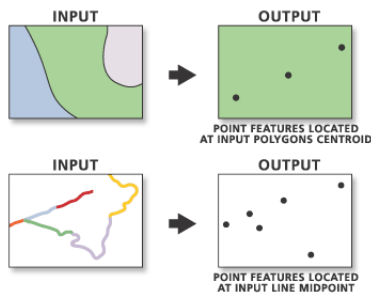
`FeatureToLine <in_features; in_features...> <out_feature_class> {cluster_tolerance} {ATTRIBUTES | NO_ATTRIBUTES}`



- 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 features 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 based on an input polygon, line, or multipoint feature class.

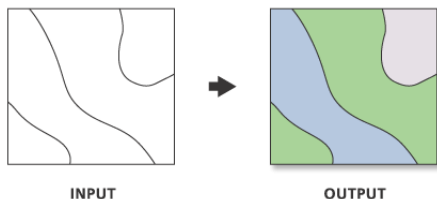
`FeatureToPoint <in_features> <out_feature_class> {CENTROID | INSIDE}`



- If the input features are polygons, the CENTROID option will result in points that are at the center of gravity for that polygon. These may not actually be inside the polygon's area. To insure 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.

Feature to Polygon: Creates a new polygon feature class from input line and/or polygon features.

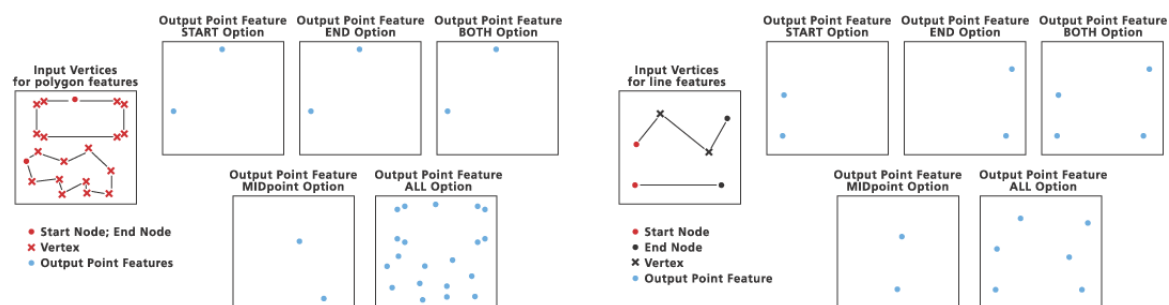
FeatureToPolygon <in_features;in_features...> <out_feature_class> {cluster_tolerance} {ATTRIBUTES | NO_ATTRIBUTES} {label_features}



- 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 a point feature class from the vertex locations of the input features.

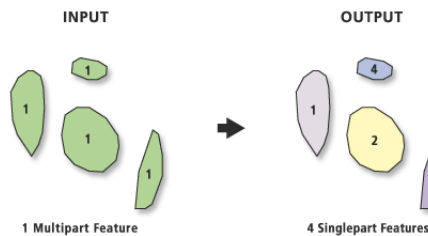
FeatureVerticesToPoints <in_features> <out_feature_class> {ALL | MID | START | END | BOTH_ENDS}



- 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.

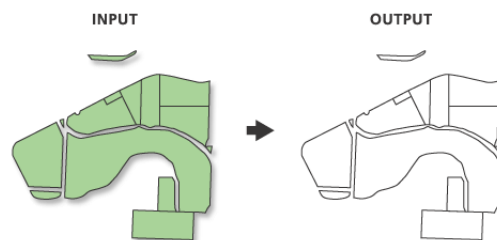
MultipartToSinglepart <in_features> <out_feature_class>



- Features that already have single-part 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: Creates a new line feature class from a polygon feature class including the feature IDs of the left and right polygons for the new lines.

`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.
- Polygon boundaries are split at nodes in the output line feature class.

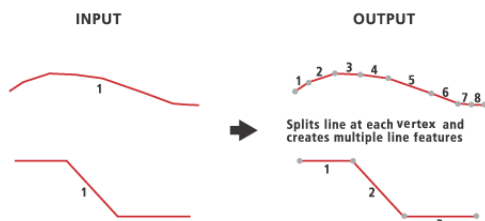
❖ **Repair Geometry:** Repairs geometry problems in a feature class or layer.

`RepairGeometry <in_features> {DELETE_NULL | KEEP_NULL}`

- 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.
- Line features that are m aware will not be modified (repaired) for any of the cases above unless they are short segments or null geometry.

Split Line at Vertices: Splits each line feature at every vertex contained in a feature class.

`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 tables 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 | RASTER>
    {field_precision} {field_scale} {field_length} {field_alias} {NULLABLE | NON_NULLABLE}
    {NON_REQUIRED | REQUIRED} {field_domain}
```

- A field of type raster allows you to have a raster image as an attribute. It is stored within or alongside the geodatabase. This is helpful when a picture is the best way to describe a feature. Precision, scale, and length cannot be set for fields of type raster.
- Coverages, stand-alone tables, feature classes from ArcSDE and personal or file 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.
- The Field Length parameter is only applicable on fields of type text or BLOB.
- For coverages, shapefiles, and dBASE tables, if field type defines a character, blanks are inserted for each record. If field type defines a numeric item, zeros are inserted for each record.
- For geodatabases, if field type defines a character or number, <null> is inserted into each record if the FieldsNullable parameter default is accepted.
- A shapefile does not support alias for fields, so you cannot add a field alias to a shapefile.

❖ **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 End Date:** Populates the values for a specified end date field with values calculated using the start date field specified.

```
CalculateEndDate <input_table> <unique_ID_fields;unique_ID_fields...> <start_date_field>
    <end_date_field>
```

- The value for the last row of the end date field will be the same as the value set for the last row of the start date field.
- This tool is useful when the intervals between start date field values are not regular and you want to animate the feature class or table through time or some other value using the Animation toolbar.
- To use this tool, the start date field must be able to be sorted in ascending order. To test this, open the attribute table for the feature class, right-click the field, and click Sort Ascending. If the field cannot be sorted in ascending order, the field must be reformatted before using this tool.

❖ **Calculate Field:** Calculates the values of a field for a feature class, feature layer, or raster catalog using an expression.

```
CalculateField <in_table> <field> <expression> {VB | PYTHON | PYTHON_9.3} {code_block}
```

- 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 of a feature class, feature layer, or raster catalog.

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

- Delete Field can be used with any table; ArcSDE, file, 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.

❖ **Transpose Time Fields:** Shifts fields that have time as a field name and their values from columns to rows in a table or feature class.

`TransposeTimeFields <input_feature_class_or_table> <fields_to_transpose;fields_to_transpose...> <output_feature_class_or_table> <time_field_name> <value_field_name> {attribute_fields;attribute_fields...}`

Input table

OID	TimeField1	TimeField2	TimeField3
0	1	2	3
1	10	20	30

Output table

OID	Time	Value
0	TimeField1	1
0	TimeField2	2
0	TimeField3	3
1	TimeField1	10
1	TimeField2	20
1	TimeField3	30

- If you want the output to be a table, you need to specify the input as a table.
- Shapefile is not a supported format for an output feature class. The output must be a geodatabase feature class.
- Object ID (such as OID, FID, and so on) and shape fields should not be set as attribute fields.



File Geodatabase toolset

Contains tools that convert individual file geodatabase vector feature classes and tables into a compressed format.

❖ **Compress File Geodatabase Data:** Converts a file geodatabase feature class from standard to compressed format.

`CompressFileGeodatabaseData <in_data>`

- Once compressed, a feature class or table is read-only and cannot be edited or modified in any way, except for changing its name and modifying attribute indexes and metadata. Compression is ideally suited to mature datasets that do not require further editing. However, if required, a compressed dataset can always be uncompressed to return it to its original, read-write format.
- You can compress a geodatabase, feature dataset, stand-alone feature class, or table. When you compress a geodatabase, all feature classes and tables within it compress. When you compress a feature dataset, all its feature classes compress.
- You cannot individually compress or uncompress a feature class in a feature dataset to produce a mixed state where some feature classes are compressed and others are not.

❖ **Uncompress File Geodatabase Data:** Converts a file geodatabase feature class or table from compressed to standard format.

`UncompressFileGeodatabaseData <in_data> {configuration_keyword}`

- You can uncompress a geodatabase, feature dataset, stand-alone feature class, or table. When you uncompress a geodatabase, all feature classes and tables within it uncompress. When you uncompress a feature dataset, all its feature classes uncompress.

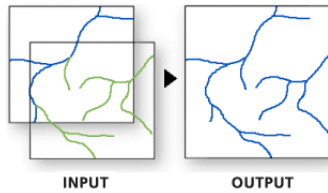
- You cannot individually compress or uncompress a feature class in a feature dataset to produce a mixed state where some feature classes are compressed and others are not. Compressed feature datasets allow you to add an uncompressed feature class to the feature dataset through operations such as creating a new, empty feature class; copying and pasting; and importing. However, once you've finished adding feature classes, you should recompress or uncompress the feature dataset so that all its feature classes are either compressed or uncompressed.

General toolset

Contains tools allowing for aggregation of features and provides simple semantic changes.

❖ **Append:** Combines multiple input datasets into an already existing target dataset.

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



- Input datasets can be point, line, or polygon feature classes, tables, rasters, or raster catalogs.
- 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).

Calculate Value: Returns a value based on a user-specified Python expression. Now available for all license levels

❖ `CalculateValue <expression> {code_block} {Variant | Address Locator | Address Locator Style | Analysis cell size | Any value | ArcMap Document | Areal unit | Boolean | CAD Drawing Dataset | Cadastral Fabric | Catalog Root | Cell Size | Composite Layer | Compression | Coordinate System | Coordinate Systems Folder | Coverage | Coverage Feature Class | Data Element | Database Connections | Dataset | Date | dBASE Table | Decimate | Disk Connection | Double | Envelope | Evaluation Scale | Extent | Feature Class | Feature Dataset | Feature Layer | Feature Set | Field | Field Info | Field Mappings | File | Folder | Formulated Raster | GeoDataServer | Geodataset | Geometric Network | Geostatistical Layer | Geostatistical Value Table | GlobeServer | GPServer | Group Layer | Horizontal factor | Image Service | Index | INFO Expression | INFO Item | INFO Table | Interop Destination Dataset | Interop Source Dataset | Layer | Layer File | Line | Linear unit | Long | M Domain | MapAlgebra Expression | MapServer | Neighborhood | Network Analyst Class FieldMap | Network Analyst Hierarchy Settings | Network Analyst Layer | Network Dataset | Network Dataset Layer | Point | Polygon | Projection File | Pyramid | Radius | Random Number Generator | Raster Band | Raster Catalog | Raster Catalog Layer | Raster Dataset | Raster Layer | Raster Statistics | Record Set | Relationship Class | Remap | Route Measure Event Properties | Schematic Dataset | Schematic Diagram | Schematic Folder | Schematic Layer | Semivariogram | Shapefile | Spatial Reference | SQL Expression | String | Table | Table View | Terrain Layer | Text File | Tile Size | Time configuration | TIN | Tin Layer | Tool | Toolbox | Topo features | Topology | Topology Layer | Vertical factor | VPF Coverage | VPF Table | WCS Coverage | weighted Overlay Table | weighted Sum | workspace | XY Domain | Z Domain}`

- Expressions can be created in a standard Python format only. Other scripting languages are not supported.

❖ **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 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 workspace supports the output data type. For example, do not try to copy a shapefile into a geodatabase.

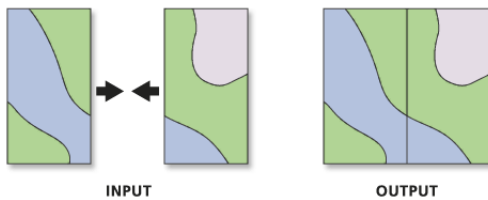
❖ **Delete:** Deletes feature datasets, feature classes, rasters, or tables.

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 if 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 on 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 data type of the first input field (of that name) it encounters. The data type may 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.

❖ **Merge Branch:** Merges two or more logical branches into a single output.

MergeBranch {in_values; in_values...}

- This tool is intended for use only in ModelBuilder.
- The tool looks at the list of inputs variables and returns the first variable that is in the has been run state.
- This tool outputs a variant that can be used as the feedback variable to any other input data element in the model.

❖ **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.

SelectData <in_data_element> <out_data_element>

- The Select Data tool is intended for use only in ModelBuilder, not at the command line.
- 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.

Aggregate Polygons: Combines disjoint and adjacent polygons into new area features based on a distance.

AggregatePolygons <in_features> <out_feature_class> <aggregation_distance>
{minimum_area} {minimum_hole_size} {NON_ORTHOGONAL | ORTHOGONAL}



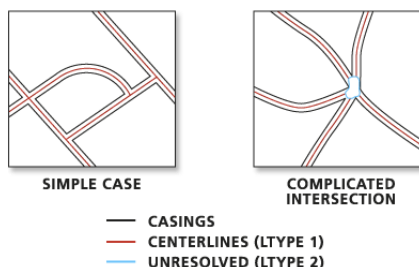
A) Nonorthogonal features

B) Orthogonal features

- The aggregation will only happen where two polygon boundaries are within the specified aggregation distance to each other. There will be no self-aggregation, meaning no aggregation within a polygon itself where one area of its boundary is less than the distance to another or between two parts of a multipart polygon.
- The output feature class will not contain any geographic attributes from the source features. A one-to-many relationship table with the name of output_feature_class_Tbl will be created that links the aggregated polygons to their source polygons. This table will contain two fields, OUTPUT_FID and INPUT_FID, which hold the aggregated object IDs and their input object IDs, respectively. With this link, you can derive necessary attributes for the output features using other geoprocessing tools. The link can become incorrect when either the input or output polygons are modified.
- The output may contain overlapping polygons and self-crossing boundaries and some connecting zones may be too narrow; therefore, further inspection and editing may be needed.

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

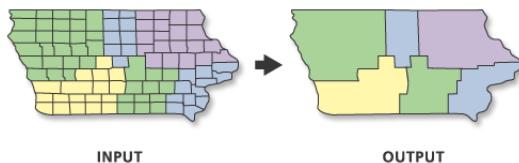
CollapseDualLinesToCenterline <in_features> <out_feature_class> <maximum_width>
{minimum_width}



- This tool is designed to work with fairly regular, near parallel pairs of lines, such as road casings. Centerlines will be created only between open-ended lines, not inside closed lines, which are likely street blocks.
- Centerlines will be derived based on the specified width parameters. A dual-line feature wider than the maximum width or narrower than the minimum width will not be centerlined. Use known road widths if available. However, since casings usually become wider at intersections, set the maximum width slightly greater than the known maximum width. You may need to experiment to find suitable parameters.
- Since the tool is not designed for natural features, such as irregularly shaped rivers, you may get unexpected results if you apply it to such features.

❖ **Dissolve:** Aggregates features based on specified attributes.

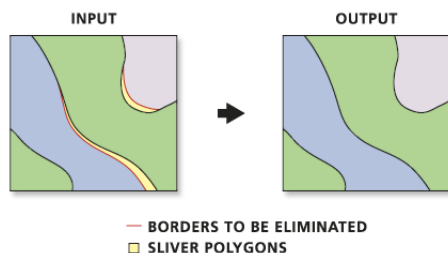
```
Dissolve <in_features> <out_feature_class> {dissolve_field;dissolve_field...}
{field{Statistics_Type}; field{Statistics_Type}...} {MULTI_PART | SINGLE_PART}
{DISSOLVE_LINES | UNSPLIT_LINES}
```



- 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: Merges the selected polygons with neighboring polygons with the largest shared border or the largest area.

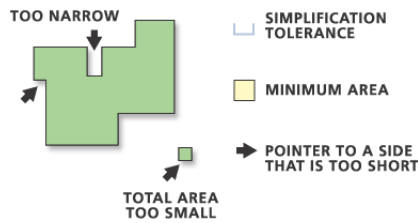
```
Eliminate <in_features> <out_feature_class> {LENGTH | AREA}
```



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

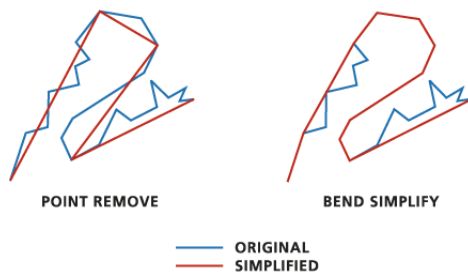
```
SimplifyBuilding <in_features> <out_feature_class> <simplification_tolerance>
{minimum_area} {NO_CHECK | CHECK_CONFLICTS}
```



- The output feature class will carry all the input fields.

◆ **Simplify Line:** Removes small fluctuations or extraneous bends from a line in a feature class while preserving its essential shape.

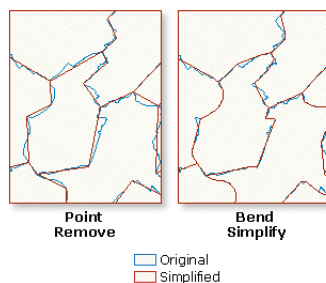
`SimplifyLine <in_features> <out_feature_class> <POINT_REMOVE | BEND_SIMPLIFY> <tolerance> {NO_CHECK | PRESERVE_SHARED | RESOLVE_ERRORS} {KEEP_COLLAPSED_POINTS | NO_KEEP}`



- The tool will produce two output feature classes, a line feature class, and a point feature class. The line output will store all the simplified lines, and the point output will store points, if any, representing closed lines that are collapsed to zero length as the result of simplification and cannot be written to the line output.
- The line output will carry all the input fields. The point output will not.

Simplify Polygon: Simplifies a polygon by removing small fluctuations or extraneous bends from its boundary while preserving its essential shape.

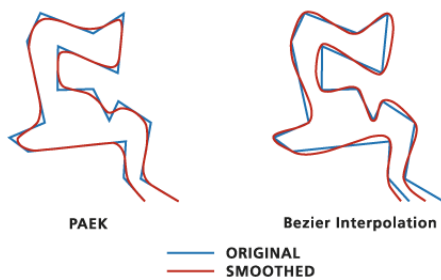
`SimplifyLine <in_features> <out_feature_class> <POINT_REMOVE | BEND_SIMPLIFY> <tolerance> {minimum_area} {NO_CHECK | PRESERVE_SHARED | RESOLVE_ERRORS} {KEEP_COLLAPSED_POINTS | NO_KEEP}`



- The tool will produce two output feature classes, a polygon feature class, and a point feature class.
- The polygon output will carry all the input fields. The point output will not.

◆ **Smooth Line:** Smooths a line to improve its aesthetic or cartographic quality.

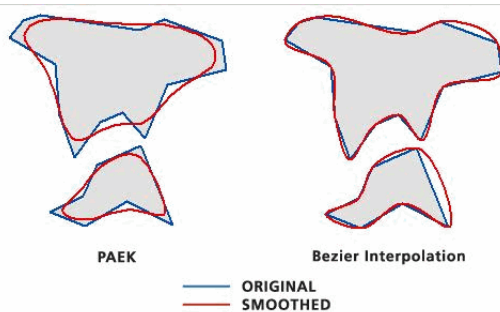
`SmoothLine <in_features> <out_feature_class> <PAEK | BEZIER_INTERPOLATION> <tolerance> {FIXED_CLOSED_ENDPOINT | NO_FIXED} {NO_CHECK | FLAG_ERRORS}`



- 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 type zero in place of the smooth_tolerance.

Smooth Polygon: Smooths a polygon to improve its aesthetic or cartographic quality.

`SmoothPolygon <in_features> <out_feature_class> <PAEK | BEZIER_INTERPOLATION> <tolerance> {FIXED_ENDPOINT | NOT_FIXED} {NO_CHECK | FLAG_ERRORS}`



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 shapefile, file geodatabase feature class, or an ArcSDE feature class.

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

- This command will only work with shapefiles, feature classes in an ArcSDE geodatabase, or feature classes in a file 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 shapefile, file geodatabase feature class, or ArcSDE feature class.

`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.

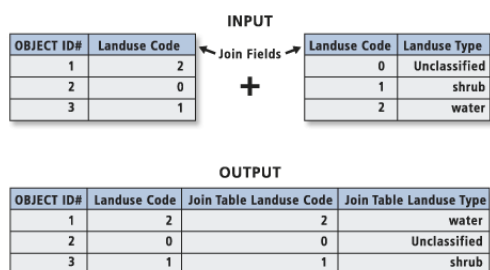


Joins toolset

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.

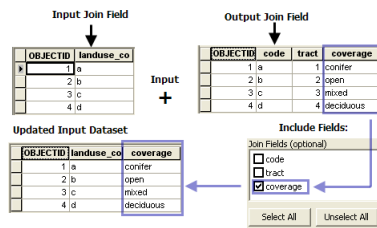
`AddJoin <in_layer_or_view> <in_field> <join_table> <join_field> {KEEP_ALL | KEEP_COMMON}`



- 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 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 (ArcSDE, file 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.

❖ **Join Field:** Joins the contents of a table to another table based on a common attribute field, while adding only user-specified fields. This join will permanently update the input dataset.

`JoinField <in_data> <in_field> <join_table> <join_field> {fields;fields...}`



- All fields in the Input Dataset will be kept during the join. Optionally, only selected fields from the Join Table will be added to the output. These can be checked under the Join Fields parameter.
- If no fields are selected for the optional Join Fields parameter, the default is to add all fields from the Join Table to the output.
- The input can be a feature class, table, or shapefile.
- Records from the Join Table can be matched to more than one record in the Input Dataset.

❖ **Remove Join:** Removes a join from a feature layer or table view.

`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.
- When a layer is joined to two tables and the first join is removed, then both joins will be removed. For example, Layer1 is joined to TableA. Then Layer1 is joined to TableB. If the join to TableA is removed, the join to TableB is also removed.

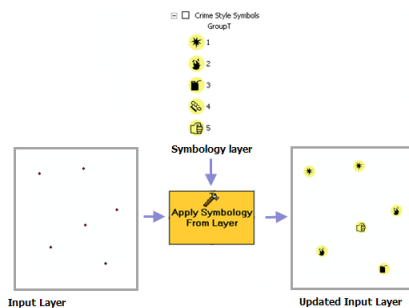


Layers and Table Views toolset

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

❖ **Apply Symbology from Layer:** Applies the symbology from a user-specified layer to the input layer.

`ApplySymbologyFromLayer <in_layer> <in_symbology_layer>`



- This tool can be applied to feature, raster, and TIN layer files or existing layers in the ArcMap table of contents.
- The symbology layer must match the type of the input layer (e.g. a feature layer cannot be applied to a raster layer and vice versa).
- Symbology from a feature layer can only be applied to features of the same geometry (e.g. a point layer cannot be applied to a polygon layer).

❖ **Make Feature Layer:** Creates a temporary feature layer from an input feature class or layer file.

`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 tool does not accept data from file geodatabases.
- 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 Image Server Layer:** Creates a raster layer from an image service.

```
MakeImageServerLayer <in_image_service> <out_image_server_layer> {template} {ID;ID...}
{mosaic_method} {order_field} {order_base_value} {lock_raster_id}
```

- The layer that is created by the tool is temporary and will not persist after the session ends unless it is saved.
- The output can be the entire image service or a portion of it.
- The output can be created with only a subset of the bands. This will help save on time and disk space.
- The mosaic_method options are only available when the image service contains an image service created with ArcGIS Image Server.

❖ **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 that will be available to select as a variable while working in the same ArcMap or ArcCatalog session.

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

- To make your layer permanent, right-click the layer in the ArcMap table of contents and click Save As Layer File or use the Save To Layer File tool.

❖ **Make Raster Layer:** Makes a temporary raster dataset layer that will be available to select as a variable while working in the same ArcMap or ArcCatalog session.

```
MakeRasterLayer <in_raster> <out_raster_layer> {where_clause} {envelope}
{Index;Index...}
```

- To make your layer permanent, right-click the layer in the ArcMap table of contents and click Save As Layer File or use the Save To Layer File tool.
- You can indicate what raster bands from the input should be used by defining an index value(s).

❖ **Make Table View:** Creates a temporary table view from an input table or feature class.

```
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 WCS Layer:** Creates an image layer from a WCS service.

```
MakewCSLayer <in_wcs_coverage> <out_wcs_layer> {template} {ID;ID...}
```

- The layer that is created by the tool is temporary and will not persist after the session ends unless the document is saved.

- The output can be the entire image service or a portion of it.
- The output can be created with only a subset of the bands. This will help save on time and disk space.

❖ **Make XY Event Layer:** Creates a temporary point layer based on x and y coordinates from a source table.

`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 an output layer file that references geographic data stored on disk.

`SaveToLayerFile <in_layer> <out_layer>`

- The input layer can be an in-memory layer created by the Make Feature Layer tool, a layer file stored on disk, or a feature layer in ArcMap.
- The feature layer or layer file may have a subset of records and fields.

❖ **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.
- 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.

❖ **Select Layer by Location:** Creates, updates, or removes a selection on the input layer based on spatial relationships to select features.

`SelectLayerByLocation <in_layer> {INTERSECT | WITHIN_A_DISTANCE | CONTAINS | COMPLETELY_CONTAINS | CONTAINS_CLEMENTINI | WITHIN | COMPLETELY_WITHIN | WITHIN_CLEMENTINI | ARE_IDENTICAL_TO | BOUNDARY_TOUCHES | SHARE_A_LINE_SEGMENT_WITH | CROSSED_BY_THE_OUTLINE_OF | HAVE_THEIR_CENTER_IN | 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.

❖ **Create Custom Geographic Transformation:** Creates a transformation method for converting data between two geographic coordinate systems or datums.

`CreateCustomGeoTransformation <geo_transformation_name> <in_coordinate_system> <out_coordinate_system> <custom_geo_transformation>`

- The output of this tool can be used as a transformation method for any tool with a parameter that requires such a method.
- All custom geographic transformation files are saved with a .gtf extension.
- Custom transformation files can't be edited. To update the file you must create a new custom geographic transformation and overwrite the existing file.
- Any geoprocessing tool that uses geographic transformations will look at all custom transformations in the default storage location and present them as valid transformation options on the dialog box (for example, the Project tool).

❖ **Define Projection:** Records the coordinate system information for the specified input dataset or feature class including any associated projection parameters, datum, and spheroid.

`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 your input feature classes or feature datasets, including the datum or spheroid, and defines the location of points on a planar surface.

`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 a spatial reference object for use in ModelBuilder and scripting.

`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 your input dataset or feature class to a new output dataset or feature class with the newly defined coordinate system, including the datum and spheroid.

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

- 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 Project tool first.



Raster (Projections and Transformations) toolset

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

- ❖ **Flip:** Reorients the raster by turning it over, from top to bottom, along the horizontal axis through the center of the raster.

`Flip <in_raster> <out_raster>`



↓ Flip



- Flip flips the grid from top to bottom along the horizontal axis through the center of the region.
- This may be useful to correct raster datasets that are upside down.

- ❖ **Mirror:** Reorients the raster by flipping it, from left to right, along the vertical axis through the center of the raster.

`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:** Transforms a raster dataset from one projection to another.

`ProjectRaster <in_raster> <out_raster> <out_coordinate_system> {NEAREST | BILINEAR | CUBIC | MAJORITY} {cell_size} {geographic_transform;geographic_transform...} {registration_point} {in_coordinate_system}`

- 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.

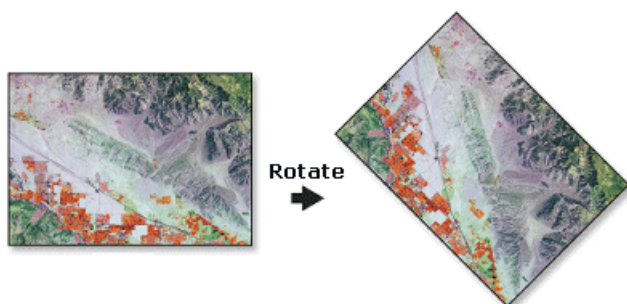
- ❖ **Rescale:** Scales a raster by the specified x and y scale factors.

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

- The scale factor must be a positive number.
- 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:** Turns the raster around the specified pivot point by angle specified in degrees.

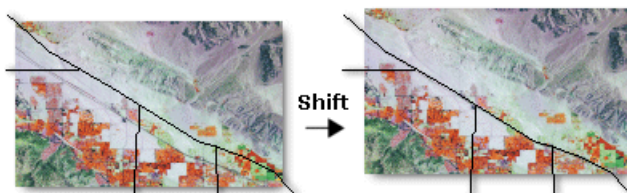
`Rotate <in_raster> <out_raster> <angle> {pivot_point} {NEAREST | BILINEAR | CUBIC | MAJORITY}`



- 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.

❖ **Shift:** Slides the raster to a new geographic location based on x and y shift values.

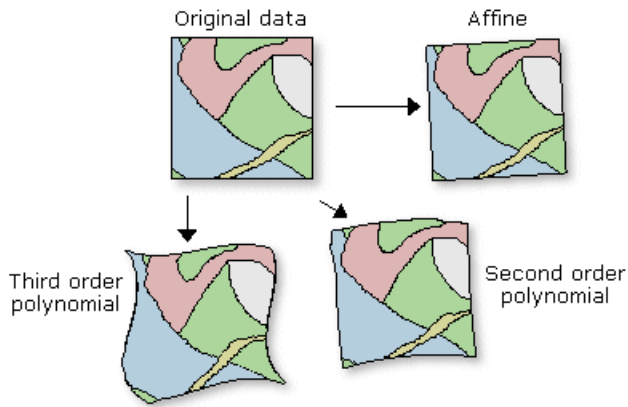
`Shift <in_raster> <out_raster> <x_value> <y_value> {in_snap_raster}`



- This tool is helpful if your raster dataset needs to be shifted to align with another data file.
- 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 | ADJUST | SPLINE} {NEAREST | BILINEAR | CUBIC | MAJORITY}`



- 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 you use more than the minimum number of links.



Raster toolset

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



Raster Catalog (Raster) toolset

Contains tools to create or work with raster catalogs.

- ❖ **Copy Raster Catalog Items:** Makes a copy of a raster catalog including all its contents or a subset of its contents if there is a selection.

```
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 comprise a geodatabase raster catalog.
- If your raster catalog output is to an ArcSDE geodatabase, a configuration keyword can be set.
- File and personal geodatabases may have one spatial grid. ArcSDE geodatabases can have up to three spatial grids.

- ❖ **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} {template_raster_catalog; template_raster_catalog...}
```

- Raster datasets within raster catalogs in a geodatabase can either be 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.

❖ **Delete Raster Catalog Items:** Deletes the raster catalogs contained in a raster catalog.

`DeleteRasterCatalogItems <in_raster_catalog>`

- Only geodatabase raster catalogs are valid inputs.

❖ **Export Raster Catalog Paths:** Creates a table listing the raster catalog paths contained in an unmanaged raster catalog. The table can be created to show all the raster catalog paths or only the broken ones.

`ExportRasterCatalogPaths <in_raster_catalog> <BROKEN | ALL> <out_table>`

- This tool can only be used for unmanaged raster catalogs.
- The output of this tool is a table, either in a geodatabase or a .DBF file.

❖ **Repair Raster Catalog Paths:** Repairs broken raster catalog paths or deletes broken links, within an unmanaged raster catalog.

`RepairRasterCatalogPaths <in_raster_catalog> <FIX | REMOVE> {original_path} {new_path}`

- You need to know the file path of the original raster catalog item location in order to change it. The Export Raster Catalog Paths tool can help you retrieve the original path names.

❖ **Workspace to Raster Catalog:** Loads all the raster datasets that are stored within the same workspace into an existing raster catalog.

`WorkspaceToRasterCatalog <in_workspace> <in_raster_catalog> {NONE | INCLUDESUBDIRECTORIES} {NONE | PROJECT_ONFLY}`

- The raster catalog must already exist for this tool to run.
- By default, the spatial reference and geotransformation of the raster datasets are persisted in the raster catalog. If you want to project the raster datasets to the spatial reference of the raster column in the raster catalog during loading, choose the Project On-the-fly option.



Raster Dataset (Raster) toolset

Contains tools to create or work with raster datasets.

❖ **Copy Raster:** Converts a raster to a BMP, GIF, GRID, IMG, JPEG, JPEG2000, PNG, TIFF, or a geodatabase raster dataset and can be used to load raster datasets into a raster catalog.

`CopyRaster <in_raster> <out_raster_dataset> {configuration_keyword} {background_value} {nodata_value} {NONE | ONEBITTO8BIT} {NONE | COLORMAPTORG} {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}`

- If you want to load raster datasets into a raster catalog, you will need to type out the full pathname of the raster catalog as the output location or drag and drop the raster catalog to the output location.

❖ **Create Random Raster:** Creates a random raster based on a user-specified distribution and extent.

`CreateRandomRaster <out_path> <out_name> <distribution> <raster_extent> <cellsize>`

- The uniform, integer, normal, and exponential distribution processing times are independent to their arguments while the poisson, gamma, binomial, geometric, and pascal distribution processing times can vary considerably when arguments are changed.

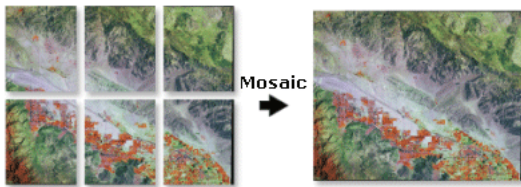
❖ **Create Raster Dataset:** Creates an empty raster dataset in a geodatabase.

```
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}
```

- 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.

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

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



- 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 raster datasets of different resolutions, it is recommended you 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).

❖ **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} {FIRST | REJECT | LAST | MATCH}
```

- The input rasters are all the raster datasets you would like to mosaic together. The inputs must have the same number of bands; otherwise, the Mosaic tool used in the model will not run.

❖ **Raster Catalog To Raster Dataset:** Mosaics a raster catalog into a raster dataset.

```
RasterCatalogToRasterDataset <in_raster_catalog> <out_raster_dataset> {where_clause}
{LAST | FIRST | MINIMUM | MAXIMUM | MEAN | BLEND} {FIRST | REJECT | LAST | MATCH} {order_by_
field} {NONE | ASCENDING} {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} {NONE
| COLOR_BALANCING} {NONE | STATISTIC_MATCHING | HISTOGRAM_MATCHING | LINEARCORRELATION_
MATCHING} {CALCULATE_FROM_ALL | SPECIFY_OID | DEFINE_FROM_SELECTION} {OID}
```

- You must set the pixel type to match your existing input raster datasets. If you do not set the pixel type, the 8-bit default will be used and your output might turn out incorrectly.
- You can save your output to BMP, GIF, GRID, IMG, JPEG, JPEG 2000, PNG, TIFF, or any geodatabase raster dataset.
- The overlapping areas of the mosaic can be handled in several ways; for example, you can set the tool to keep only the first raster dataset's data, or you can blend the overlapping cell values. There are also several options to determine how to handle a colormap, if the raster dataset uses one. For example, you can keep the colormap of the last raster dataset used in the mosaic.

- Color matching and color correction can be used to make the raster mosaic more seamless.

❖ **Workspace to Raster Dataset:** Creates a mosaicked raster dataset from all the raster datasets that are stored within the same workspace.

`WorkspaceToRasterDataset <in_workspace> <in_raster_dataset> {NONE | INCLUDE_SUBDIRECTORIES} {LAST | FIRST | BLEND | MEAN | MINIMUM | MAXIMUM} {FIRST | REJECT | LAST | MATCH} {background_value} {nodata_value} {NONE | ONEBITTO8BIT} {mosaicking_tolerance} {NONE | STATISTIC_MATCHING | HISTOGRAM_MATCHING | LINEARCORRELATION_MATCHING}`

- The target raster dataset must already exist for the tool to run. If a raster does not already exist, use the Create Raster Dataset tool to create a new raster dataset.
- All raster datasets within the specified workspace will be mosaicked into the target raster dataset.
- Since mosaicking will take place, you will need to specify the mosaic method and color map mode to use.

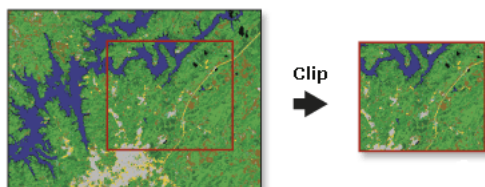


Raster Processing (Raster) toolset

Contains tools used in processing raster datasets.

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

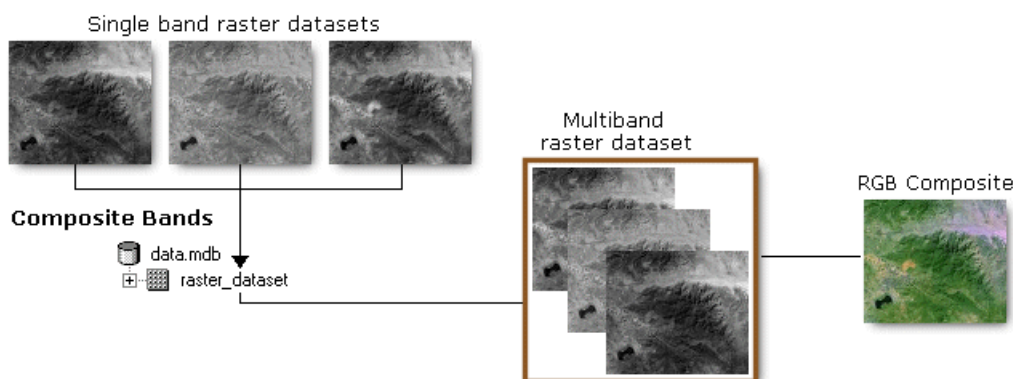
`Clip <in_raster> <rectangle> <out_raster> {in_template_raster} {nodata_value} {NONE | ClippingGeometry}`



- 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 single raster dataset of multiple bands from multiple single-band raster datasets.

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



- The output raster dataset takes the cell size from the first raster band in the list.
- By default, the output raster dataset takes the extent and the spatial reference of the first raster band with a spatial reference in the list. You can change this by setting Output Extent and Output Coordinate System in the Environment Settings.

- ❖ **Create Ortho-Corrected Raster Dataset:** Create an orthorectified raster dataset using the rational polynomial coefficients (RPC) associated with a raster dataset.

```
CreateOrthoCorrectedRasterDataset <in_raster> <out_raster_dataset> <CONSTANT ELEVATION | DEM> <constant_elevation> <in_DEM_raster> {ZFactor} {ZOffset} {NONE | GEOID}
```

- To orthorectify a raster dataset, the raster must have RPCs associated with it.
- A DEM can be used in the orthophoto rectification process so that the elevation and curvatures of the earth can be taken into account.
- If a DEM is used to orthorectify the raster dataset, the constant elevation value will not be used.

- ❖ **Create Pan-Sharpener Raster Dataset:** Fuses a high-resolution panchromatic raster dataset with a lower resolution raster dataset to create an RGB raster with the resolution of the panchromatic raster.

```
CreatePanSharpenedRasterDataset <in_raster> <red_channel> <green_channel> <blue_channel> {infrared_channel} <out_raster_dataset> <in_panchromatic_image> <ESRI | IHS | Brovey | SIMPLE MEAN> {red_weight} {green_weight} {blue_weight} {infrared_weight}
```

- Only the areas that fully overlap will be affected by this tool.
- IHS uses intensity, hue, and saturation color space for data fusion. Brovey uses an algorithm based on spectral modeling for data fusion.

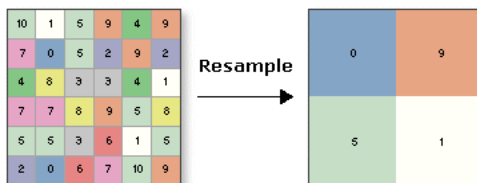
- ❖ **Extract Subdataset:** Extracts raster datasets stored within a subdataset raster file.

```
ExtractSubDataset <in_raster> <out_raster> {ID; ID...}
```

- Subdataset file formats can be either HDF4 or NITF files.
- Each subdataset can have multiple bands associated with it.
- If you do not choose to specify one or more subdatasets, the first subdataset will be returned.
- You can save your output to BMP, GIF, GRID, IMG, JPEG, JPEG 2000, PNG, TIFF, personal geodatabase, file geodatabase, or ArcSDE geodatabase raster formats.
- The GIF format does not support multiband; therefore, it is not a valid output format unless your raster dataset is single band.

- ❖ **Resample:** Changes the cell size of a grid.

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



- 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.



Raster Properties (Raster) toolset

Contains tools to modify raster dataset properties.

- ❖ **Add Colormap:** Allows you to add a color map to a raster dataset, if it does not already exist.

`AddColormap <in_raster> <in_template_raster>`

- This tool allows you to either create or edit a color map table for a raster dataset.
- The color map, from a raster dataset, that already exists will be applied to the input raster dataset.

- ❖ **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.

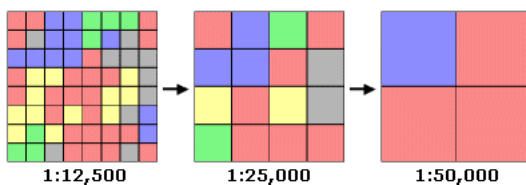
- ❖ **Batch Calculate Statistics:** Allows you to calculate statistics on multiple raster datasets.

`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.

- ❖ **Build Raster Attribute Table:** Adds a raster attribute table to a raster dataset or updates an existing one.

`BuildRasterAttributeTable <in_raster> {NONE | OVERWRITE}`

- If you want to delete an existing table and create a new one, check the overwrite check box. A new raster attribute table will be created.
- If you have an existing table and you do not specify OVERWRITE, the table will be updated. No fields will be deleted, but the values in the table will be up to date.

- ❖ **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 1 will use each pixel and a value of 2 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.

❖ **Delete Colormap:** Removes a color map that is associated with a raster dataset.

`DeleteColormap <in_raster>`

❖ **Delete Raster Attribute Table:** Removes the raster attribute table that is associated with a raster dataset.

`DeleteRasterAttributeTable <in_raster>`

- The input raster dataset can only have a single band.

❖ **Export Raster World File:** Creates a world file based on the geographic information of a raster dataset. The pixel size and the location of the upper left pixel are extracted for the world file.

`ExportRasterWorldFile <in_raster_dataset>`

- The output world file will depend on the file format being used.
- If the transformation cannot be expressed as a world file, the tool will write an approximate affine transformation into the world file, with an 'x' on the end of the extension name.

❖ **Get Cell Value:** Retrieves the pixel value at a specific x,y coordinate.

`GetCellValue <in_raster> <location_point> {ID;ID...}`

- For multi-band raster datasets you can specify from which bands to retrieve the cell value. If you do not specify any bands, the pixel value for all the bands will be returned for that location.
- This tool is useful when you need to acquire pixel value from within a geoprocessing model. In ArcMap, you can use the Identify tool instead.

❖ **Get Raster Properties:** Returns the properties of a raster dataset.

`GetRasterProperties <in_raster> {MINIMUM | MAXIMUM | MEAN | STD | UNIQUEVALUECOUNT | TOP | LEFT | RIGHT | BOTTOM | CELLSIZE | CELLSIZEY | VALUETYPE | COLUMNCOUNT | ROWCOUNT | BANDCOUNT}`

- The property returned will be displayed in the geoprocessing window.



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 fields or features in the origin table and the destination table.

`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.

- ❖ **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 ownership.



Subtypes toolset

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

- ❖ **Add Subtype:** Adds a new subtype to the subtypes in the input 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 input table using its code.

```
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 (RDBMS) statistics of business tables, feature tables, and delta tables along with the statistics of those tables' indexes.

`Analyze <in_dataset> <components>;components...>`

- 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.
- The components are BUSINESS, FEATURE, RASTER, ADDS, and DELETES.

- ◆ **Change Privileges:** Establishes or changes user access privileges to the input ArcSDE dataset, standalone feature class, or table.

`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 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:** Writes the rows from an input table, table view, feature class, or feature layer to a new table.

`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.
- If the input rows are from a feature class or table, then all rows will be used. If the input rows are from a layer or table view which has a selection, only the selected features or rows will be used.
- If the output table already exists, it will be overwritten.

- ❖ **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 or file 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.
- Coverages, CAD, and VPF data are invalid input to this tool. Rows cannot be deleted from these data types without destroying the integrity of the data.
- If there is no selection in the input rows, all rows 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 the input table fields, based on the selected pivot field and value field, in the output table 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

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.
- This new feature class can be empty or may contain existing features. It must be in the same feature dataset as the topology.
- 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 drop-down 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 to 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}
```

- Neither personal nor file geodatabases 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> {PRIVATE | PUBLIC | PROTECTED}
```

- The output version name is prefixed by the ArcSDE 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 a version against a parent version in its lineage.

```
ReconcileVersion <in_workspace> <version_name> <target_name> {BY_OBJECT | BY_ATTRIBUTE} {FAVOR_TARGET_VERSION | FAVOR_EDIT_VERSION} {LOCK_AQUIRED | NO_LOCK_AQUIRED} {NO_ABORT | ABORT_CONFLICTS} {NO_POST | POST}
```

- 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> {NO_EDITS_TO_BASE | EDITS_TO_BASE}
```

- File and personal geodatabases don't support versioning. Versioning tools only work with ArcSDE data.

- 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 an ArcSDE dataset as versioned in ArcCatalog.

`UnregisterAsVersioned <in_dataset> {KEEP_EDIT | NO_KEEP_EDIT} {NO_COMPRESS_DEFAULT | COMPRESS_DEFAULT}`

- Making a feature class or table multiversioned requires a unique integer field. Only the owner of the data may 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 first.



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}`

❖ **Create File GDB:** Creates a new file geodatabase.

`CreateFileGDB <out_folder_path> <out_name>`

- Each dataset can scale up to 1 TB in size.

❖ **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.

NOTE: This tool is no longer supported at ArcGIS 9.2, because indexes do not need to be automated. Use the Create Address Locator tool to create a new address locator.

`AutomateGeocodingIndexes <in_address_locator>`

❖ **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> {config_keyword}`

- Address locators can be created in the same workspace of the reference data or any other workspace you specified.
- The role of 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.

NOTE: This tool is no longer supported at ArcGIS 9.2, because indexes do not need to be deautomated. Use the Create Address Locator tool to create a new address locator.

`DeautomateGeocodingIndexes <in_address_locator>`

❖ **Delete Address Locator:** Deletes an address locator.

NOTE: This tool is no longer supported at ArcGIS 9.2. Use the Delete (Data Management) tool to delete an address locator.

`DeleteAddressLocator <in_address_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 workspace.

❖ **Rebuild Address Locator:** Rebuilds an address locator.

`RebuildAddressLocator <in_address_locator>`

- You must have write privileges on the address locator to use this tool.
- The reference feature class and tables must be available. You may need to repair the address locator particularly the paths to the reference data if the data has been moved to a different location since the last time the locator was built.
- Address locators created based on a versioned geodatabase will be rebuilt using the same version.
- Address locators created with versions prior to ArcGIS 9.2 or address locators provided by any StreetMap™ products cannot be rebuilt.

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

NOTE: This tool is no longer supported at ArcGIS 9.2, because address attributes and indexes in the address locator will be refreshed. Use the Rebuild Address Locator tool to rebuild an address locator.

`RebuildGeocodingIndex <in_address_locator>`

❖ **Rematch Addresses:** Rematches addresses in a geocoded feature class.

`RematchAddresses <in_geocoded_feature_class> {in_where_clause}`

- The input feature class has to be a feature class that was created by the Geocode Addresses tool or a process where a table of addresses were geocoded.

❖ **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 workspace, 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:** Recalculates route measures using points.

```
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 object ID field, use Make Query Table prior to Dissolve Route Events to add a virtual object ID 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 feature layer using routes and route events.

```
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.

Mobile toolbox

Contains tools to convert data to formats optimized and ready for use within ArcGIS Mobile applications.

❖ **Create Mobile Basemap:** Creates a basemap dataset that is optimized and ready for use with ArcGIS Mobile applications.

```
CreateMobileBasemap <in_map_document> <out_folder> {in_data_frame} {extract_extent}  
{extract_boundary} {in_layer;in_layer...}
```

- The input ArcMap map document should contain references to all data layers (operational and basemap) to be used by the mobile device. Operational data refers to data that will be synchronized between a mobile device and a server; basemap data refers to data used for reference or orientation purposes only.
- This command only supports the conversion of vector data; any raster layers referenced in the input map document are not eligible for extraction.
- Annotation layers are not supported.
- The output folder must exist prior to command execution.
- Only data from one data frame can be extracted at a time. If the in_data_frame parameter is not used then layers from the active data frame will be extracted.

❖ **Generate Mobile Service Cache:** Generates a mobile service cache for use with ArcGIS Mobile applications by extracting selected operational vector and basemap raster layers from a mobile data access enabled ArcGIS Server Map Service.

```
GenerateMobileServiceCache <map_service> <out_folder> {override_extract_extent} {in_  
layer;in_layer...} {in_version} {NO_ERASE | ERASE}
```

- Only ArcGIS Server Map Services which have been published with mobile data access capabilities can be used by this command.
- The selected ArcGIS Server Map Service should contain references to all operational vector layers and basemap raster layers to be used by the mobile device. Operational data refers to data that will be synchronized between a mobile device and a server; basemap data refers to data used for reference or orientation purposes only.
- Only vector layers based on enterprise geodatabase data sources can be synchronized between the mobile device and the ArcGIS Server.
- The output folder must exist prior to command execution.
- By default, the extent that is extracted is based on the map services' initial map extent; this can be overridden by using the override_extract_extent parameter.

Multidimension toolbox

Contains tools to make a netCDF raster layer, feature layer, or table view; to convert to netCDF from raster, feature, or table; and to select a dimension of a netCDF layer or table to display.

❖ **Feature To NetCDF:** Converts a feature class to a netCDF file.

```
FeatureToNetCDF <in_features> <field {variable} {Units};field {Variable} {Units}...> <out_netCDF_file> {field {Dimension} {Units};field {Dimension} {Units}...}
```

- The default variable name is the same as the field name specified in the field_to_variable parameter.
- The type of variable is the same as the type of field.
- The default dimension name is the same as the field name specified in the field_to_dimension parameter.
- The size of a dimension is equal to the number of unique values in the respective field.

❖ **Make NetCDF Feature Layer:** Makes an in-memory feature layer from a netCDF file.

```
MakeNetCDFFeatureLayer <in_netCDF_file> <variable;variable...> <x_variable> <y_variable>  
<out_feature_layer> {row_dimension;row_dimension...} {z_variable} {m_variable}  
{dimension {value};dimension {value}...} {BY_VALUE | BY_INDEX}
```

- 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.
- An existing feature layer will be overwritten if the same layer name is specified.

❖ **Make NetCDF Raster Layer:** Makes an in-memory raster layer from a netCDF file.

```
MakeNetCDFRasterLayer <in_netCDF_file> <variable> <x_dimension> <y_dimension>  
<out_raster_layer> {band_dimension} {dimension {value};dimension {value}...}  
{BY_VALUE | BY_INDEX}
```

- To create a netCDF raster layer from a netCDF variable, the spacing between x-coordinates must be equal and the spacing between y-coordinates must be equal. If the coordinates are unequally spaced, create a netCDF feature layer, then interpolate to raster.
- The output raster layer type is either float or integer based on the netCDF variable type.
- The first variable in the netCDF file suitable for creating a raster is selected as the default variable.

❖ **Make NetCDF Table View:** Makes a table view from a netCDF file.

```
MakeNetCDFTableView <in_netCDF_file> <variable;variable...> <out_table_view>  
{row_dimension;row_dimension...} {dimension {value};dimension {value}...}  
{BY_VALUE | BY_INDEX}
```

- Table views are tables stored in memory and are the same as the table view when a table is added to ArcMap.
- 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 exists, the tables in memory are removed.
- Table views created in ArcCatalog cannot be used in ArcMap.
- An existing table view will be overwritten if the same table view name is entered.

❖ **Raster to NetCDF:** Converts a raster dataset to a netCDF file.

```
RasterToNetCDF <in_raster> <out_netCDF_file> {variable} {variable_units} {x_dimension}
{y_dimension} {band_dimension} {field {Dimension} {Units};field {Dimension} {Units}...}
```

- The input can be any valid raster dataset or raster catalog.
- The default variable name is the same as the input raster name.
- The output netCDF variable type is either float or integer based on the input raster dataset type.

❖ **Select by Dimension:** Updates the netCDF layer display or netCDF table view based on the dimension value.

```
SelectByDimension <in_layer_or_table> {dimension {Value};dimension {Value}...}
{BY_VALUE | BY_INDEX}
```

- Inputs for this tool can be created by using the Make NetCDF Feature Layer, Make NetCDF Raster Layer, or Make NetCDF Table View tools.
- If a dimension is not specified, its value is set to the first value. The first value is considered as the default value.
- This tool updates the input. In ModelBuilder, an output variable appears to chain the updated input as input to another tool in the model, but the tool does not produce a new output.

❖ **Table to NetCDF:** Converts a table to a netCDF file.

```
TableToNetCDF <in_table> <field {Variable} {Units};field {Variable} {Units}...>
<out_netCDF_file> {field {Dimension} {Units};field {Dimension} {Units}...}
```

- The default variable name is the same as the field name specified in the field_to_variable parameter.
- The type of the variable is the same as the type of the field.

Server toolbox

Contains statistical tools for analyzing the distribution of geographic features.

Caching toolset

- ❖ **Create Map Server Cache:** Creates the cache tiling scheme for a given map service. Once the tiling scheme is created use the Manage Map Server Cache Tiles tool to generate tiles for your cache.

```
CreateMapServerCache <server_name> <object_name> <data_frame> <out_folder> <NEW |  
PREDEFINED> <STANDARD | CUSTOM> <number_of_scales> <dpi> <tile_width> <tile_height>  
{FUSED | MULTI_LAYER} {tiling_schema} {tile_origin} {Scale;Scale...} {Layer;Layer...}  
{NONE | ANTIALIASING} {PNG8 | PNG24 | PNG32 | JPEG} {tile_compression_quality}
```

- This tool only works with ArcGIS Server map services.
- Only one data frame can be cached at a time. If other data frames are needed, separate caches must be generated that associated with separate map services.
- Once you create the tiling scheme you cannot modify it. You can however add or delete scales to an existing tiling scheme using the Manage Map Server Cache Scales tool.
- The Number of Scales parameter is the number of different map scales the tool will create layers for in the cache.
- JPEG is best used with raster data. When used with vector data, lines and text may be blurred.
- The cache tile format cannot be changed once the cache is generated. The cache must first be deleted before switching to a different file format.

- ❖ **Delete Globe Server Cache:** Deletes an existing globe service layer's cache.

```
DeleteGlobeServerCache <server_name> <object_name> <Layer;Layer...>
```

- This is an unrecoverable operation so only use if you are sure you no longer need the cache.
- By default all layers of the service are displayed and selected. If you intend to delete only a particular layers cache, make sure to unselect the ones you want to keep the cache for.
- Delete Globe Server Cache tool can be helpful if you have many caches residing in your server cache directory and want to delete only a particular services cache.
- Delete Globe Server Cache deletes the entire cache folder. Note that after Delete Globe Server Cache tool runs it will restart the server object. This will in turn generate a new set of caches for each layer in the service that was deleted by the delete operation. This is because all globe services will require a cache configuration to exist on disk to run. The cache that gets created automatically on start up of a service is a skeleton representation that does not contain any tiles.

- ❖ **Delete Map Server Cache:** Deletes an existing map service cache including all associated files on disk.

```
DeleteMapServerCache <server_name> <object_name> <data_frame> <Layer;Layer...>
```

- This is an unrecoverable operation so only use if you are sure you no longer need the cache
- If there is more than one data frame in the cache directory only the active data frame of the specified map service will be removed.
- If there is only one data frame in the cache directory the entire directory will be removed.
- After this tool runs the map service will be restarted.

❖ **Generate Globe Server Cache:** Generates globe data caches based on the ArcGlobe data tiling scheme.

NOTE: This tool has been deprecated in ArcGIS 9.3 and replaced with the **Manage Globe Server Cache Tiles** tool.

```
GenerateGlobeServerCache <server_name> <object_name> <out_folder> <GLOBE - 1:10000000 |
CONTINENT - 1:5000000 | COUNTRIES - 1:2500000 | COUNTRY - 1:1250000 | STATES - 1:625000 | STATE
- 1:312500 | COUNTIES - 1:156250 | COUNTY - 1:78125 | METROPOLITAN AREA - 1:39062 | CITIES -
1:19531 | CITY - 1:9765 | TOWN - 1:4882 | NEIGHBORHOOD - 1:2441 | CITY BLOCKS - 1:1220 | CITY
BLOCK - 1:610 | BUILDINGS - 1:305 | BUILDING - 1:152 | HOUSES - 1:76 | HOUSE PROPERTY - 1:38 |
HOUSE - 1:19 | ROOMS - 1:9 | ROOM - 1:4> <ROOM - 1:4 | ROOMS - 1:9 | HOUSE - 1:19 | HOUSE PROPERTY
- 1:38 | HOUSES - 1:76 | BUILDING - 1:152 | BUILDINGS - 1:305 | CITY BLOCK - 1:610 | CITY BLOCKS -
1:1220 | NEIGHBORHOOD - 1:2441 | TOWN - 1:4882 | CITY - 1:9765 | CITIES - 1:19531 | METROPOLITAN
AREA - 1:39062 | COUNTY - 1:78125 | COUNTIES - 1:156250 | STATE - 1:312500 | STATES - 1:625000
| COUNTRY - 1:1250000 | COUNTRIES - 1:2500000 | CONTINENT - 1:5000000 | GLOBE - 1:10000000>
{thread_count} <Layer;Layer...>
```

❖ **Generate Map Server Cache:** Generates a cache of static image tiles for an ArcGIS Server map service.

NOTE: This tool has been deprecated in ArcGIS 9.3 and replaced with the **Create Map Server Cache** tool, followed by the **Manage Map Server Cache Tiles** tool.

```
GenerateMapServerCache <server_name> <object_name> <data_frame> <out_folder> <NEW |
PREDEFINED> <STANDARD| CUSTOM> <num_of_scales> <dpi> <tile_width> <tile_height> {FUSED|
MULTI_LAYER} {tiling_schema} {tile_origin} {Scale; Scale...} {Layer; Layer...} {thread_
count} {ANTIALIASING | NONE} {PNG8 | PNG24 | PNG32 | JPEG} {tile_compression_quality}
```

❖ **Generate Map Server Cache Tiling Scheme:** Generates a tiling scheme xml file for use when creating a static map service cache.

```
GenerateMapServerCacheTilingScheme <map_document> <data_frame> <tile_origin>
<tiling_schema> <cache_levels> <Scale;Scale...> <dpi> <tile_width> <tile_height>
```

- By default, the tiling origin starts at the upper left of the coordinate reference system of the data source.
- Once a cache has been created with a tiling scheme it cannot use a different tiling scheme in the future (while updating, for example).

❖ **Manage Globe Server Cache Tiles:** Updates an existing globe service cache to replace missing tiles, overwrite outdated tiles, or add new tiles in new areas.

```
ManageGlobeServerCacheTiles <server_name> <object_name> <layer {level_from} {level_to};layer
{level_from} {level_to}...> <RECREATE EMPTY TILES | RECREATE ALL TILES> {update_extent}
{thread_count} {update_feature_class} {IGNORE_COMPLETION_STATUS_FIELD | TRACK_COMPLETION_
STATUS}
```

- Running update without specifying an extent will update the entire extent of the service being for the specified levels of detail.
- Update is useful to run when you need to update only a portion of the globe services cache. When specifying the from and to levels of detail, make sure you specify all levels you want tiles generated for. The level_from defines the lowest level of detail you'd want your data cache to begin with. The level_to defines the highest resolution you'd want your data caching to have.

❖ **Manage Map Server Cache Scales:** Update map server cache scales on an existing cached map service.

```
ManageMapServerCacheScales <server_name> <object_name> <Scale;Scale...>
```

- Use this tool to add new scales or delete existing scales from a map cache.
- Once you define a tiling schema for a map cache you cannot add new scales to it. Use this tool to insert new cache scales or remove existing cache scales.
- If you remove scales from an existing cache, it will permanently delete all existing cached tiles within that level of detail.

- ❖ **Manage Map Server Cache Tiles:** Creates new tiles or replaces missing tiles, overwrites outdated tiles, or adds new tiles in new areas by specific extents or using a polygon feature class to generate tiles by feature extents, or, in the case of a multi-layer cache, from additional layers.

```
ManageMapServerCacheTiles <server_name> <object_name> <data_frame> <Layer;Layer...>
<Levels;Levels...> <RECREATE EMPTY TILES | RECREATE ALL TILES | DELETE TILES> {constraining_
extent} {thread_count} {NONE | ANTIALIASING} {update_feature_class} {IGNORE_COMPLETION_
STATUS_FIELD | TRACK_COMPLETION_STATUS}
```

- Use Recreate Empty Tiles mode to add tiles in an extent not previously cached.
- Use Recreate All Tiles to update outdated tiles.
- Before running this tool, configure the map service to use as many instances as possible. This will dramatically decrease cache update time.

- ❖ **Update Globe Server Cache:** Updates an existing globe service cache to replace missing tiles, overwrite outdated tiles, or add new tiles in new areas.

NOTE: This tool is has been deprecated in ArcGIS 9.3 and replaced with the Manage Globe Server Cache Tiles tool.

```
UpdateGlobeServerCache <server_name> <object_name> {update_extent} <layer;layer...> <LOD_
from> <LOD_to> {thread_count} <Recreate Empty Tiles | Recreate All Tiles>
```

- ❖ **Update Map Server Cache:** Updates an existing map service cache to replace missing tiles, overwrite outdated tiles, or add new tiles in new areas or, in the case of a multi-layer cache, from additional layers.

NOTE: This tool is has been deprecated in ArcGIS 9.3 and replaced with the Manage Map Server Cache Tiles tool.


```
UpdateMapServerCache <server_name> <object_name> <data_frame> <layer;layer...>
{constraining_extent} <scale;scale...> <RECREATE_EMPTY_TILES| RECREATE_ALL_TILES>
{thread_count} {NONE | ANTIALIASING} {update_feature_class} {CACHE ALL FEATURES AND IGNORE
COMPLETION STATUS FIELD | TRACK CACHE COMPLETION STATUS FOR EACH FEATURE}
```


Spatial Statistics toolbox

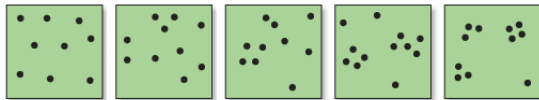
Contains statistical tools for analyzing the distribution of geographic features.

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} {area}`

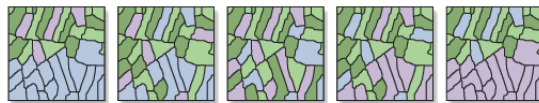


Dispersed ← → Clustered

- 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 the degree of clustering for either high values or low values.

`HighLowClustering <input_feature_class> <input_field> {FALSE | TRUE} <INVERSE DISTANCE | INVERSE DISTANCE SQUARED | FIXED DISTANCE BAND | ZONE OF INDIFFERENCE | POLYGON CONTIGUITY (FIRST ORDER) | GET SPATIAL WEIGHTS FROM FILE> <EUCLIDEAN DISTANCE | MANHATTAN DISTANCE> <NONE | ROW> <distance_band_or_threshold_distance> {weights_matrix_file}`




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 tool (Inside option) to create centroids before performing the nearest neighbor operation.

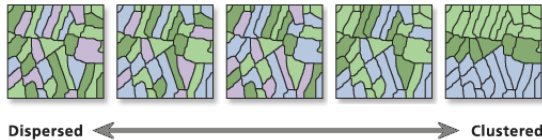
-  **Multi-Distance Spatial Cluster Analysis (Ripley's K-function):** Determines whether a feature class is clustered at multiple different distances. The tool outputs the result as a table and optionally as a pop-up graphic.

`MultiDistancespatialClustering <input_feature_class> <output_table> <EUCLIDEAN DISTANCE | MANHATTAN DISTANCE> <number_of_distance_bands> {0 PERMUTATIONS - NO CONFIDENCE ENVELOPE | 9 PERMUTATIONS | 99 PERMUTATIONS | 999 PERMUTATIONS} {FALSE | TRUE} {weight_field} {beginning_distance} {distance_increment} {NONE | SIMULATE OUTER BOUNDARY VALUES | REDUCE ANALYSIS AREA | RIPLEY'S EDGE CORRECTION FORMULA} {MINIMUM ENCLOSING RECTANGLE | USER PROVIDED STUDY AREA FEATURE CLASS} {study_area_feature_class}`

- The output of the tool is a table with two fields named “ExpectedK” and “ObservedK” containing the expected k and observed k values, respectively. If a confidence interval option is specified, two additional fields named “LowConfEnv” and “HiConfEnv” will be present with the confidence interval information for each iteration of the tool.
- The confidence envelope is constructed by distributing points randomly in the study area and calculating k for that distribution. Each random distribution of the points is known as a “permutation.” If 99 permutations is selected, the tool will randomly distribute the set of points 99 times for each iteration. After distributing the points 99 times, the tool selects the k value that deviated above and below the expected by the greatest amount and these values become the confidence interval.
- The number of points distributed for each permutation is equal to the number of points in the input feature class. If the user has specified a weight field, the weight of the random points will be taken randomly from a list of the weights of the input data.

 **Spatial Autocorrelation (Morans 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 | POLYGON CONTIGUITY (FIRST ORDER) | GET SPATIAL WEIGHTS FROM FILE> <EUCLIDEAN DISTANCE | MANHATTAN DISTANCE> <NONE | ROW> <distance_band_or_threshold_distance> {weights_matrix_file}`



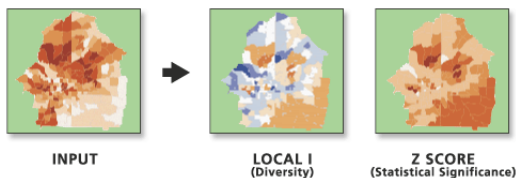
- Given a set of features and an associated attribute, Global Moran’s I evaluates whether the pattern expressed is clustered, dispersed, or random. A Moran’s I value near +1.0 indicates clustering; a value near –1.0 indicates dispersion.

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 Morans I):** Identifies those 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 | POLYGON CONTIGUITY (FIRST ORDER) | GET SPATIAL WEIGHTS FROM FILE> <EUCLIDEAN DISTANCE | MANHATTAN DISTANCE> <NONE | ROW> <distance_band_or_threshold_distance> {weights_matrix_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.

 **Hot Spot Analysis (Getis-Ord Gi*):** Calculates the Getis-Ord Gi* statistic for hot spot analysis.


HotSpot <input_feature_class> <input_field> <output_feature_class> <FIXED DISTANCE BAND | INVERSE DISTANCE | INVERSE DISTANCE SQUARED | ZONE OF INDIFFERENCE | POLYGON CONTIGUITY (FIRST ORDER) | GET SPATIAL WEIGHTS FROM FILE> <EUCLIDEAN DISTANCE | MANHATTAN DISTANCE> <NONE | ROW> {distance_band_or_threshold_distance} {self_potential_field} {weights_matrix_file}



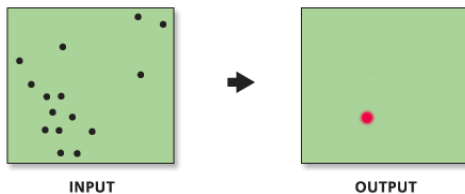
- Given a set of weighted data points, the Getis-Ord Gi* statistic identifies those clusters of points with values higher in magnitude than you might expect to find by random chance. (For line and polygon features, centroids are calculated prior to analysis.)
- The Gi function creates a new feature class 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.

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.

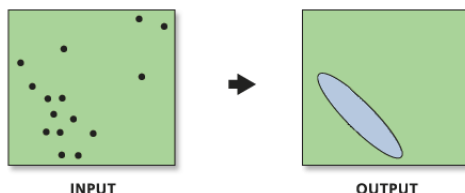
CentralFeature <input_feature_class> <output_feature_class> <EUCLIDEAN DISTANCE | MANHATTAN DISTANCE> {weight_field} {self_potential_weight_field} {case_field}




- The feature associated with the smallest accumulated distance to all other features 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).
- For line and polygon features, feature centroids are used in the computations.
- Calculations are based on either Euclidean or Manhattan distance and require projected data to accurately measure distances.

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

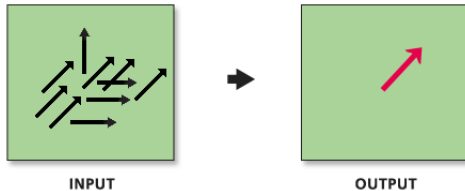
DirectionalDistribution <input_feature_class> <output_ellipse_feature_class> <1 STANDARD DEVIATION | 2 STANDARD DEVIATIONS | 3 STANDARD DEVIATIONS> {weight_field} {case_field}




- 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, YStdDist stores the standard distance in the y direction, and Rotation 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 lines.

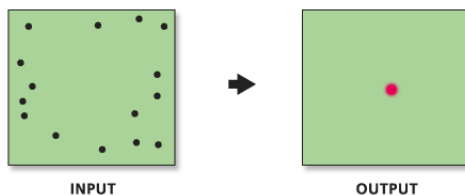
`DirectionalMean <input_feature_class> <output_feature_class> <orientation_only> {case_field}`




- The input feature class must be a line or polyline feature class.
- 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 zero). When input vector directions span the entire compass, the circular variance is large (near 1).
- Calculations are based on either Euclidean or Manhattan distance and require projected data to accurately measure distances.

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

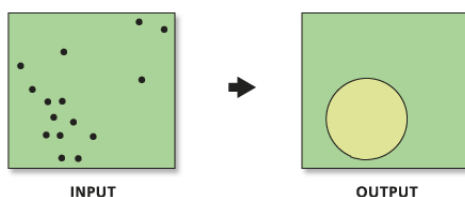
`MeanCenter <input_feature_class> <output_feature_class> {weight_field} {case_field} {dimension_field}`



- The mean center is a point constructed from the average x- and y-values for the input feature centroids.
- 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}`



- The standard distance calculation may be based on an optional weight (to get the standard distance of businesses weighted by employees, for example).
- The standard distance is a useful statistic, as it provides a single summary measure of feature distribution around any given point (similar to the way a standard deviation measures the distribution of data values around the statistical mean).
- Standard Distance creates a new feature class containing a circle polygon centered on each of the central features (mean centers). Each circle polygon is drawn with a radius equal to the standard distance. The attribute value for each circle polygon is its standard distance.
- If the underlying pattern in the data is truly random, the one standard deviation polygon will cover approximately 68 percent of the features in the cluster. Two standard deviations will contain approximately 95 percent of the features, and three standard deviations will cover approximately 99 percent of the features in the cluster.
- A mean center calculation is used to determine the geographic centers of the input feature class.



Modeling Spatial Relationships toolset

Contains tools to construct spatial weights matrices or model spatial relationships using regression analyses.



Generate Network Spatial Weights: Constructs a spatial weights matrix file (.swm) from a network dataset; which defines spatial relationships among all features in terms of an underlying network structure. Requires a Network Analyst license.

```
GenerateNetworkSpatialWeights <input_feature_class> <unique_ID_field> <output_spatial_weights_matrix_file> <input_network> <impedance_attribute> {impedance_cutoff} {maximum_number_of_neighbors} {barriers} {ALLOW_UTURNS | NO_UTURNS | ALLOW_DEAD_ENDS_ONLY} {Restrictions;Restrictions...} {NO_HIERARCHY | USE_HIERARCHY} {search_tolerance} {INVERSE | FIXED} {exponent} {ROW_STANDARDIZATION | NO_STANDARDIZATION}
```

- This tool was designed to work with point input feature class data only.



Generate Spatial Weights Matrix: Constructs a spatial weights matrix (.swm) file to represent the spatial relationships among features in a dataset.

```
GenerateSpatialWeightsMatrix <input_feature_class> <unique_ID_field> <output_spatial_weights_matrix_file> <INVERSE_DISTANCE | FIXED_DISTANCE | K_NEAREST_NEIGHBORS | CONTIGUITY_EDGES_ONLY | CONTIGUITY_EDGES_CORNERS | DELAUNAY_TRIANGULATION | CONVERT_TABLE> {EUCLIDEAN | MANHATTAN} {exponent} {threshold_distance} {number_of_neighbors} {ROW_STANDARDIZATION | NO_STANDARDIZATION} {input_table}
```

- This tool honors the environment output coordinate system. Feature geometry is projected to the output coordinate system prior to analysis, so values entered for the Threshold Distance parameter use the same units as those specified in the output coordinate system. All mathematical computations are based on the output coordinate system. Consequently, if the output coordinate system does not match the input feature class spatial reference, either make sure, for all analyses using the spatial weights matrix file, that the output coordinate system matches the settings used when the spatial weights matrix file was created.
- Whenever using a distance-based Conceptualization of Spatial Relationships, you should project your data using a Projected Coordinate System (rather than a Geographic Coordinate System based on degrees, minutes, and seconds) prior to analysis. To avoid confusion, this projection should match the environment output coordinate system settings.
- The unique_ID_field is used to relate features to one another (the relationship or weight between features 1 and 5, for example). Consequently, the unique_ID_field values must be unique for every feature and typically should be a permanent field that will remain with the feature class.

- ❖ **Geographically Weighted Regression (GWR):** Performs a local form of linear regression used to model spatially varying relationships.

```
GeographicallyWeightedRegression <in_features> <dependent_field> <explanatory_field>
<out_feature_class> <FIXED | ADAPTIVE> <AICC | CV | BANDWIDTH_PARAMETER> {distance}
{number_of_neighbors} {weight_field} {coefficient_raster_workspace} {cell_size} {in_
prediction_locations} {Prediction_explanatory_field;Prediction_explanatory_field...}
{out_prediction_feature_class}
```

- In global regression models, such as Ordinary Least Squares (OLS), results are unreliable when two or more variables exhibit multicollinearity (when two or more variables are redundant or together tell the same “story”). GWR builds a local regression equation for each feature in the dataset. When the values for a particular explanatory variable cluster spatially, you will very likely have problems with local multicollinearity. The condition number in the out_feature_class indicates when results are unstable due to local multicollinearity. As a rule of thumb, do not trust results for features with a condition number larger than 30.
- Caution should be used when including nominal or categorical data in a GWR model. Where categories cluster spatially, there is strong risk of encountering local collinearity issues. The condition number included in the GWR output indicates when local collinearity is a problem. Results in the presence of local collinearity are unstable.
- GWR should be applied to datasets with several hundred features for best results. It is not an appropriate method for small datasets.
- A regression model is misspecified if it is missing a key explanatory variable. Statistically significant spatial autocorrelation of the regression residuals and or unexpected spatial variation among the coefficients of one or more explanatory variables suggests that your model is misspecified. You should make every effort (through OLS residual analysis and GWR coefficient variation analysis) to discover what these key missing variables are so they may be included in the model.

- ❖ **Ordinary Least Squares:** Performs global Ordinary Least Squares linear regression to generate predictions or to model a dependent variable in terms of a its relationships to a set of explanatory variables. Requires an ArcInfo or Spatial Analyst license.

```
OrdinaryLeastSquares <input_feature_class> <unique_ID_field> <output_feature_class>
<dependent_variable> <Explanatory_variables;Explanatory_variables ...> {coefficient_
output_table} {diagnostic_output_table}
```

- The unique_ID field is used to link output results to input features. Consequently, the unique_ID field values must be unique for every feature and typically should be a permanent field that will remain with the feature class.
- The OLS model is misspecified, and consequently results from OLS regression are unreliable, whenever there is statistically significant spatial autocorrelation of the regression residuals. Be sure to run the Spatial Autocorrelation tool on your regression residuals to assess this potential problem. Statistically significant spatial autocorrelation of regression residuals usually indicates a key missing explanatory variable.
- When misspecification is the result of trying to model non-stationary variables using a global model (OLS is a global model), then Geographically Weighted Regression may be used to improve predictions and to better understand the non-stationarity (regional variation) inherent in your explanatory variables.
- You should visually inspect the over and under predictions evident in your regression residuals to see if they provide clues about potential missing variables from your regression model. It sometimes helps to run Hot Spot Analysis on the residuals to help you visualize spatial clustering of the over and under predictions.



Rendering toolset

Contains tools to perform a variety of data rendering tasks but are being phased out. With each tool description will be the recommended tool (if it exists) to replace the current rendering tool.



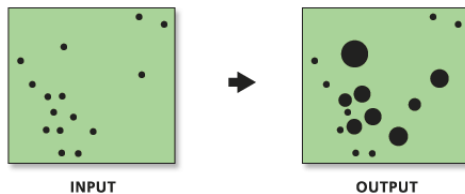
❖ **Cluster/Outlier Analysis with Rendering:** Identifies clusters of points with values similar in magnitude and clusters of points with very heterogeneous values within a given set of weighted data points and applies a “cold to hot” type of rendering. Replace with Cluster and Outlier Analysis (Anselin Local Morans I); rendering is now automatic.

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



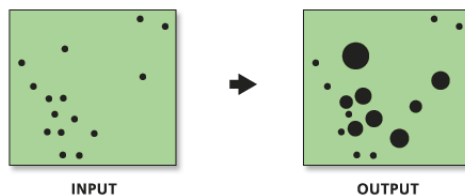
❖ **Collect Events with Rendering:** Converts event data to weighted point data and applies a graduated circle rendering to the count field. Replace with Collect Events; rendering is now automatic.

```
CollectEventsRendered <input_incident_features> <output_layer_file> <output_weighted_
point_feature_class>
```



❖ **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}
```

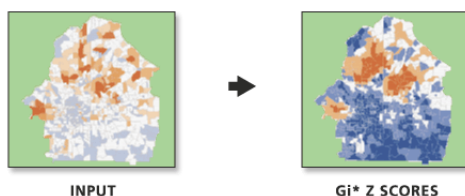



- The Count Renderer draws quantities using circle size to show relative values.



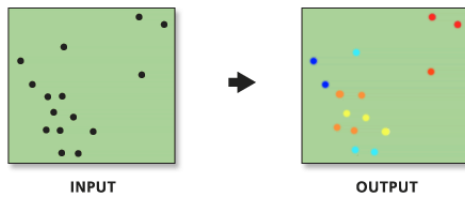
❖ **Hot Spot Analysis with Rendering:** Calculates Gi* statistics and applies a cold to hot type of rendering to the output Z scores. Replace with Hot Spot Analysis (Getis-Ord Gi*); rendering is now automatic.

```
HotSpotsRendered <input_feature_class> <input_field> <output_layer_file> <output_feature_
class> <distance_band_or_threshold_distance>
```



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


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



- This tool is appropriate for rendering output from both Hot Spot Analysis and Diversity Analysis.


Utilities toolset

Contains tools to perform a variety of data conversion tasks.

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

`CalculateAreas <input_feature_class> <output_feature_class>`

- This tool is useful in determining a weight for intrazonal interaction.
- 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.

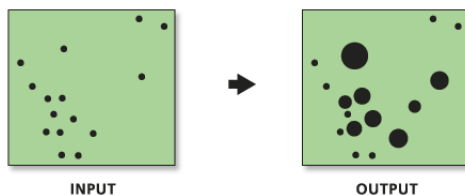
 **Calculate Distance Band from Neighbor Count:** Calculates the distance for all features in the input feature class to their nth neighbor (specified in the Neighbors parameter) and examines the distances and returns three values: the maximum distance, the minimum distance, and the average distance.

`CalculateDistanceBand <Input_Features> <Neighbors> <EUCLIDEAN DISTANCE | MANHATTAN DISTANCE>`

- This tool calculates the distance for all features in the input feature class to their nth neighbor. Then it examines the distances and returns three values: the maximum distance, the minimum distance, and the average distance.
- The neighborhood of all features will have at least n neighbors if the neighborhood size is greater than the maximum distance reported by the tool.
- The neighborhood of all features will have fewer than n neighbors if the neighborhood size is less than the minimum distance reported by the tool.

 **Collect Events:** Collects event data into weighted point data.

`CollectEvents <input_incident_features> <output_weighted_point_feature_class>`



- Collect Event creates a new feature class and adds a field named "Count" to hold the sum of all incidents for each unique location.
- This function creates as derived output the name of the count field created and the maximum count value encountered for any one location. These outputs are useful when using the tool in conjunction with a rendering tool in both the model building and scripting environments.

❖ **Convert Spatial Weights Matrix to Table:** Converts a binary spatial weights matrix file to a table.

`ConvertSpatialWeightsMatrixToTable <input_spatial_weights_matrix_file> <output_table>`

- This tool allows you to edit a spatial weights matrix file. Create a spatial weights matrix file using the Generate Spatial Weights Matrix or Generate Network Spatial Weight tools. Convert the resultant spatial weights matrix file to a table. Modify the spatial relationships as appropriate. Finally, use the Generate Spatial Weights Matrix tool to convert the modified table back to the binary spatial weights matrix file format.

❖ **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;value_field...><SPACE | COMMA | SEMI-COLON>
<output_ascii_file>`

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 a tool used to convert to and from a TIN.

Layer 3D to Feature Class: Applies 3D properties associated with a feature layer and writes the result to a new multipatch feature class.

`Layer3DToFeatureClass <in_feature_layer> <out_feature_class> {group_field}`

- For an input feature to be converted, the result must support representation as a multipatch. For example, points symbolized as 3D markers can be converted whereas extruded points (for instance, vertical lines) cannot.
- Only certain 3D properties are applied: 3D symbols assigned to points, 3D symbols assigned to lines, extrusion applied to polygons, and texture down-sampling assigned to multipatches.

From Feature Class (Conversion) toolset

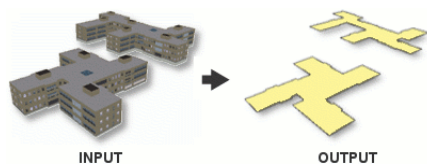
Contains tools used to convert from a feature class.

Feature Class Z to ASCII: Exports 3D points, multipoints, polylines, or polygons to ASCII text files in xyz or generate format.

`FeatureClassZToASCII <in_feature_class> <output_location> <out_file> {GENERATE | XYZ} {SPACE | COMMA} {AUTOMATIC | FIXED} {digits_after_decimal} {DECIMAL_POINT | DECIMAL_COMMA}`

MultiPatch Footprint: Converts multipatches into polygons.

`MultiPatchFootprint <in_feature_class> <out_feature_class>`



- The output polygons represent the 2D area covered by the multipatches when viewed from directly above.

From File (Conversion) toolset

Contains tools used to convert 3D features from files.

ASCII 3D to Feature Class: Imports 3D features from one or more ASCII text files into a new output feature class.

`Ascii3DToFeatureClass <input;input...> <XYZ | XYZI | GENERATE> <out_feature_class> <MULTIPOINT | POINT | POLYLINE | POLYGON> {z_factor} {input_coordinate_system} {average_point_spacing} {file_suffix} {DECIMAL_POINT | DECIMAL_COMMA}`

- This tool imports XYZ, XYZI, or generate file formats.

Import 3D Files: Imports one or more 3D models into an output feature class.

`Import3DFiles <in_files;in_files...> <out_feature_class> {ONE_ROOT_ONE_FEATURE | ONE_FILE_ONE_FEATURE} {spatial_reference} {Z_IS_UP | Y_IS_UP}`

- The output feature class must be placed in a geodatabase, rather than a shapefile, if textures (images mapped on geometry faces) are to be preserved.
- GeoVRML is the only format that has a defined coordinate system. The other formats tend to use local coordinate systems (for instance, centered around 0,0,0). In the latter case, the output shapes will need to be georeferenced.
- When multiple files are input, the coordinate system of the first file is used for all the files. For all formats other than GeoVRML, this will be “Unknown.”

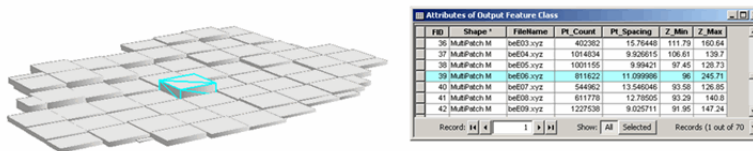
LAS to Multipoint: Imports one or more files in LAS format, the industry standard for LiDAR data, into a new multipoint feature class.

```
LASToMultipoint <input;input...> <out_feature_class> {average_point_spacing}
{class_code;class_code...} {ANY_RETURNS | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | LAST_RETURNS}
{keyword {Name};keyword {Name}...} {input_coordinate_system} {file_suffix}
```

- The output multipoint feature class must be placed in a feature dataset. This is because the points are spatially sorted into areas (tiles) whose size is determined, in part, by the extent of the feature dataset.
- If you aren't interested in importing points based on their return number or if all returns specified in the file(s) are set to zero because the points have been filtered or classified, select ANY_RETURNS.

Point File Information: Generates a new output feature class containing statistical information about one or more point files.

```
PointFileInformation <input;input...> <out_feature_class> <LAS | XYZ | XYZI | GENERATE>
<file_suffix> {input_coordinate_system} {NO_RECURSION | RECURSION} {extrude_geometry}
{DECIMAL_POINT | DECIMAL_COMMA}
```



- The statistical information presented in the feature attribute table consists of the point count, average point spacing, z minimum, and z maximum of each point file entered. A separate row is created for each input file encountered. The point spacing is an estimate that assumes the points within the input file are evenly spaced over the XY extent of each input file.
- An output polygon feature class is created with the XY extents of the input file(s). Alternatively, MultiPatch features can be generated by utilizing z minimum and z maximum extent information for each input file. These provide a 3D bounding box representation that can be viewed in ArcScene or ArcGlobe.
- When a folder containing point data files is selected as input the suffix must be entered. If a single file is selected as input the suffix is not a requirement.



From Raster (Conversion) toolset

Contains tools used to convert rasters to 3D features.

Raster Domain: Writes the interpolation zone of the input raster surface into a new 3D polyline or 3D polygon feature class.

```
RasterDomain <in_raster> <out_feature_class> <LINE | POLYGON>
```

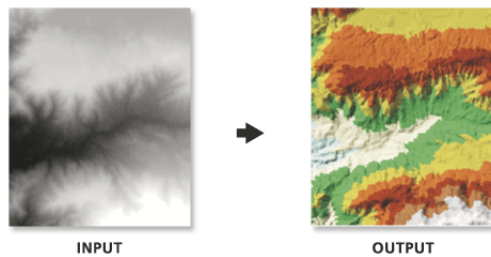


- To create a polygon feature class based on the raster surface's interpolation zone, use the POLYGON option.

- To create a polyline feature class based on the raster surface's interpolation zone, use the LINE option.

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.

From Terrain (Conversion) toolset

Contains tools used to convert from terrains.

Terrain to Raster: Converts a terrain dataset into a raster.

`TerrainToRaster <in_terrain> <out_raster> {FLOAT | INT} {LINEAR | NATURAL_NEIGHBORS} {OBSERVATIONS 250 | CELLSIZE 10} {resolution}`

- The cellsize parameter controls the horizontal resolution of the output raster.
- The resolution parameter indicates which pyramid level of the terrain to use for conversion. Pyramid levels are defined using z-tolerance, which represents the approximate vertical accuracy relative to the full resolution data.
- To extract a subset of the terrain, define the extent using the geoprocessing environment settings.

Terrain to TIN: Converts a terrain dataset to a file-based TIN.

`TerrainToTin <in_terrain> <out_tin> {pyramid_level_resolution} {max_nodes} {CLIP | NO_CLIP}`

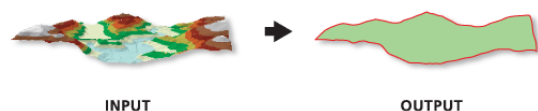
- Define the extent of the output TIN using the geoprocessing environment extent setting.
- Use an extent and pyramid level that are not likely to be too large for a TIN. While the maximum size of a TIN that can be used under Win32 is between 15 to 20 million nodes, it's recommended to cap the size at a few million. Triangulated surfaces larger than this are best represented by terrain datasets.

From TIN (Conversion) toolset

Contains tools used to convert from TINs.

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

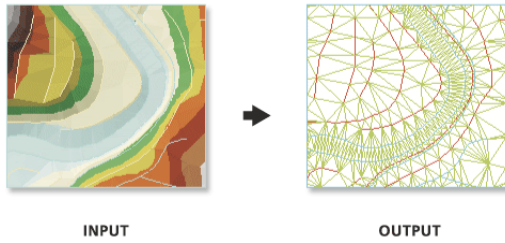
`TinDomain <in_tin> <out_feature_class> <LINE | POLYGON>`



- Produces a 3D polygon or polyline feature class.
- 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.

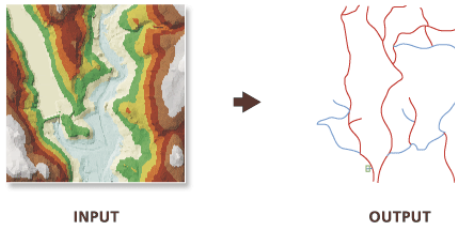
`TinEdge <in_tin> <out_feature_class> {DATA | SOFT | HARD | ENFORCED | REGULAR | OUTSIDE | ALL}`



- Produces a line feature class whose lines are extracted edges of the input TIN.
- Use the optional {edge_type} argument to extract a specific type of triangle edge.

TIN Line: Writes the hard and soft breaklines from a TIN into a new 3D polyline feature class.

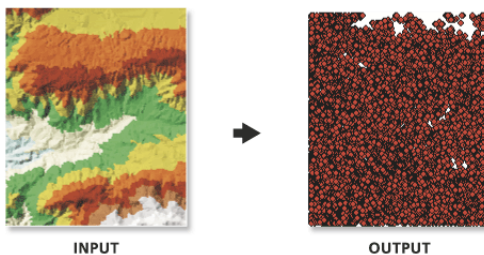
`TinLine <in_tin> <out_feature_class> {code_field}`



- The input TIN must have some breaklines for this tool to produce output line features.
- If you want to convert TIN edges into line features, regardless of whether or not they belong to breaklines, use the TIN Edge tool.

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

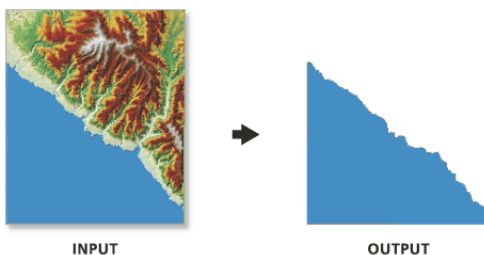
`TinNode <in_tin> <out_feature_class> {spot_field} {tag_field}`



- Produces a 2D or 3D point feature class whose points are extracted from nodes of the input TIN.
- 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.

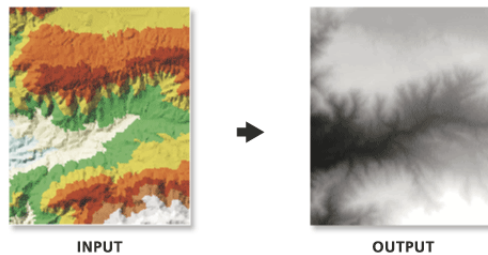
`TinPolygonTag <in_tin> <out_feature_class> {tag_field}`



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

TIN to Raster: Converts a TIN to a raster.

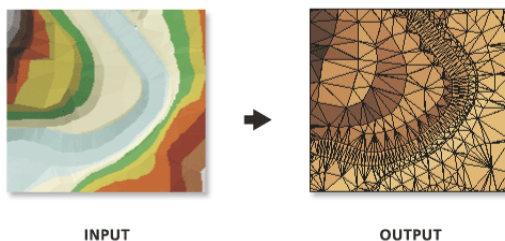
```
TinRaster <in_tin> <out_raster> {FLOAT | INT} {LINEAR | NATURAL_NEIGHBORS}
{OBSERVATIONS 250 | CELLSIZE 10} {z_factor}
```



- TIN to Raster interpolates cell z-values from the input TIN at the specified resolution or sampling interval to produce the output raster.
- 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}
```



- Produces a polygon feature class whose polygons are constructed from the input TIN's triangles.



To KML (Conversion) toolset

Contains tools used to convert to Keyhole Markup Language (KML) files.

Layer to KML: Converts an in-memory or file-based feature or raster layer into a Keyhole Markup Language (KML) file containing a translation of ESRI geometries and symbology into KML. This file is compressed using zip compression and will have a “.KMZ” extension and can be read by any KML client including ArcGIS Explorer, ArcGlobe, and Google Earth.

```
LayerToKML <layer> <out_kmz_file> <layer_output_scale> {No_COMPOSITE | COMPOSITE}
{boundary_box_extent} {image_size} {dpi_of_client}
```

- The output KMZ file cannot already exist.
- You can reduce the size of the output KMZ document if your layer has a scale-dependent renderer and you choose an appropriate “Layer Output Scale”.
- To output a single raster image draped over topography use the “Return single composite image” option.
- To output every layer as separate raster image use the “Convert Vector to Raster” option.

Map to KML: Converts an in-memory Map or Map Document into a Keyhole Markup Language (KML) file containing a translation of ESRI geometries and symbology into KML. This file is compressed using zip compression and will have a “.KMZ” extension and can be read by any KML client including ArcGIS Explorer, ArcGlobe, and Google Earth.

```
MapToKML <in_map_document> <data_frame> <out_kmz_file> <map_output_scale> {NO_COMPOSITE | COMPOSITE} {VECTOR_TO_IMAGE | VECTOR_TO_VECTOR} {extent_to_export} {image_size} {dpi_of_client}
```

- The output KMZ file cannot already exist.
- You can reduce the size of the output KMZ document if your map has scale-dependent renderers and you choose an appropriate “Map Output Scale”.
- To output a single raster image draped over topography use the “Return single composite image” option.
- To output every layer as separate raster image use the “Convert Vector to Raster” option.

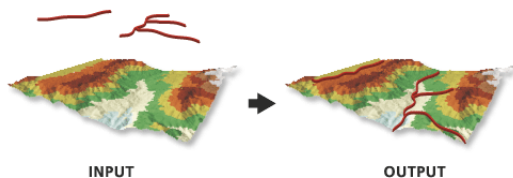


Functional Surface toolset

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 raster, TIN, or terrain surface.

```
InterpolateShape <in_surface> <in_feature_class> <out_feature_class> {sample_distance} {z_factor} {method} {DENSIFY | VERTICES_ONLY} {pyramid_level_resolution}
```



- Interpolates a 3D feature class from an input 2D feature class and a surface with the overlapping extent.
- When using the natural neighbors interpolation option, make sure to specify a reasonable sample distance. This usually falls between 0.5 and 1.0 times the average point spacing of the data used to build the TIN.
- When using the vertex only option, input features with vertices that fall outside the data area of the surface will be ignored and not available in the output. Clip features prior to running the interpolate shape to ensure the features are completely on the surface.

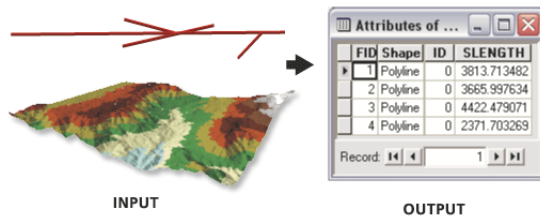
Line of Sight: Calculates the visibility across a surface between points.

```
LineOfSight <in_surface> <in_line_feature_class> <out_los_feature_class> {out_obstruction_feature_class} {NO_CURVATURE | CURVATURE} {NO_REFRACTION | REFRACTION} {refraction_factor} {pyramid_level_resolution}
```

- 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 raster or TIN surface.

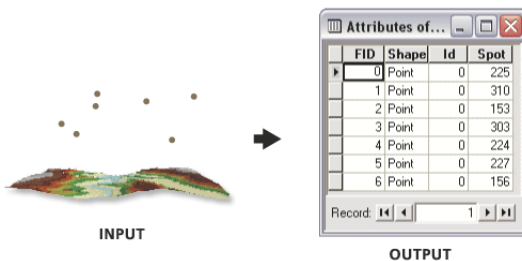
```
SurfaceLength <in_surface> <in_feature_class> {out_length_field} {sample_distance} {z_factor}
```

- 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 raster or TIN surface.

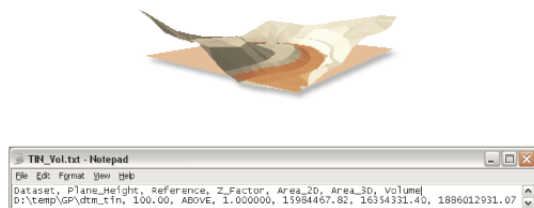
`surfaceSpot <in_surface> <in_feature_class> {out_spot_field} {z_factor} {method}`



- Ensure 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 than 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 surface from 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 have to 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.

 **Spline with Barriers:** Interpolates a surface, using barriers, from points using a minimum curvature spline technique. The barriers are entered as either polygon or polyline features.

SplinewithBarriers <in_point_features> <z_field> {in_barrier_features} <output_cell_size>
<output_raster> {smoothing_factor}

- This tool requires the Java Runtime Environment Version 5.0, or higher, to be installed. The Java Runtime Environment can be downloaded free from <http://www.java.com/en/download>.
- The resulting smooth surface from the spline with barrier tool is constrained by the input barrier features.
- If a cell size of zero is entered the shorter of the width or the height of the extent of the input point features in the input spatial reference, divided by 250, will be used as the cell size.

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 $nrows \times ncols$, 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: Interpolates a hydrologically correct surface from point, line, and polygon data using parameters specified in a file.

```
TopoToRasterByFile <in_parameter_file> <out_surface_raster> {out_stream_features} {out_sink_features}
```

- 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.\
- The parameter file is structured with the input datasets listed first, followed by the various parameter settings, then the output options.

Inputs:

- Contours—Contour line dataset with item containing height values.
`Contour <Path> <Item>`
- Points—Point dataset with item containing height values.
`Point <Path> <Item>`
- 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>.
`Sink <Path> <Item>`
- Streams—Stream line dataset. Height values are not necessary.
`Stream <Path>`
- Lakes—Lake polygon dataset. Height values are not necessary.
`Lake <Path>`
- Boundary—Boundary polygon dataset. Height values are not necessary.
`Boundary <Path>`

Parameter settings:

- Enforce—Controls whether drainage enforcement is applied.
`ENFORCE <ON | OFF | ON_WITH_SINK>`
- Datatype—Primary type of input data.
`DATATYPE <CONTOUR | SPOT>`
- Iterations—The maximum number of iterations the algorithm performs.
`ITERATIONS <#>`

- Roughness Penalty—The measure of surface roughness.
`ROUGHNESS_PENALTY <#>`
- Discretization Error Factor—The amount to adjust the data smoothing of the input data into a raster.
`DISCRETE_ERROR_FACTOR <#>`
- Vertical Standard Error—The amount of random error in the z-values of the input data.
`VERTICAL_STANDARD_ERROR <#>`
- Tolerances—The first reflects the accuracy of elevation data in relation to surface drainage, and the other prevents drainage clearance through unrealistically high barriers.
`TOLERANCES <#> <#>`
- Z-Limits—Lower and upper height limits.
`ZLIMITS <#> <#>`
- Extent—Minimum x, minimum y, maximum x, and maximum y coordinate limits.
`EXTENT <#> <#> <#> <#>`
- Cell Size—The resolution of the final output raster.
`CELL_SIZE <#>`
- Margin—Distance in cells to interpolate beyond the specified output extent and boundary.
`MARGIN <#>`

Output options:

- Output Stream Features—Only use if Output stream polyline features is set in the Topo to Raster by File dialog box.
`OUT_STREAM`
- Output Sink Features—Only use if Output remaining sink point features is set in the Topo to Raster by File dialog box.
`OUT_SINK`
- Output Diagnostics File—The location and name of the diagnostics file.
`OUT_DIAGNOSTICS <Path>`

Trend: Interpolates a surface from points using a trend technique.

`Trend <in_point_features> <z_field> <out_raster> {cell_size} {order} {LINEAR | LOGISTIC} {out_rms_file}`

- 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 root mean square (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 LOGISTIC option, the z-value field of input point features should have codes of 0 and 1.

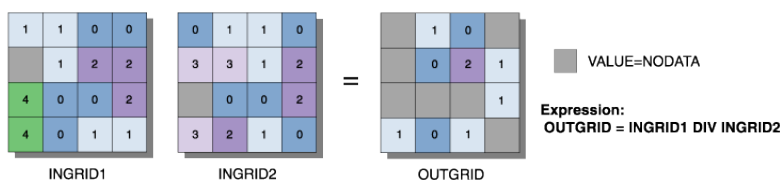


Raster Math toolset

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 zero, 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 into a floating-point value.

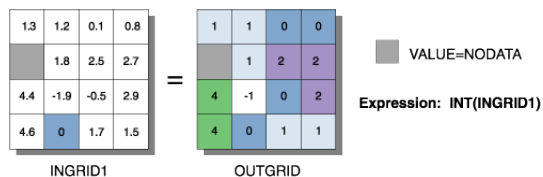
Float <in_raster_or_constant> <out_raster>



- 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>

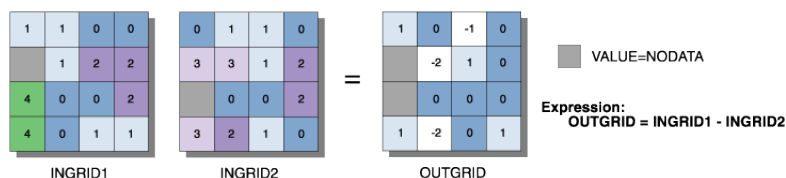


- If rounding is preferred rather than truncating, add a 0.5 input raster prior to performing Int.
- The difference between the Round Down and Int functions is that Int always truncates a number:
 - int on 1.5 becomes 1
 - int on -1.5 becomes -1
 while for the same two values, Round Down returns
 - round 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.

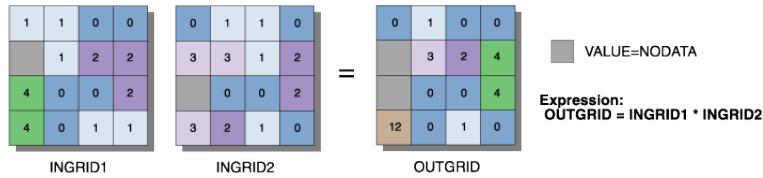
Plus <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- 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 of a raster by zones of equal interval, equal area, or by natural breaks.

`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.

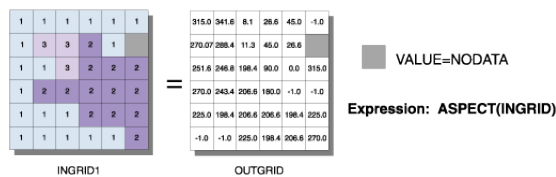


Raster Surface toolset

Contains tools to analyze the surface of a raster dataset.

Aspect: Identifies the downslope direction of the maximum rate of change in value from each cell to its neighbors.

`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 may be obtained by first performing a Neighborhood Focal Statistics operation with the mean option on the input raster.

Contour with Barriers: Creates contours from a raster surface. The inclusion of barrier features allows you to independently generate contours on either side of a barrier.

```
ContourwithBarriers <in_raster> <out_contour_feature_class> {in_barrier_features}
{POLYLINES | POLYGONS} {in_contour_values_file} {NO_EXPLICIT_VALUES_ONLY | EXPLICIT_
VALUES_ONLY} {in_base_contour} <in_contour_interval> <in_indexed_contour_interval>
<in_contour_list; in_contour_list...> {in_z_factor}
```

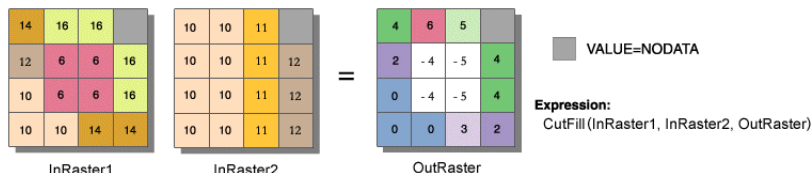
Curvature: Calculates the curvature of a raster surface, optionally including profile and plan curvature.

```
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.

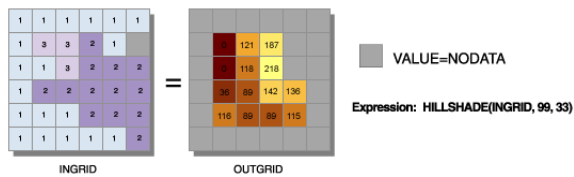
```
CutFill <in_before_surface> <in_after_surface> <out_raster> {z_factor}
```



- 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: Computes hillshade values for a raster surface 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.

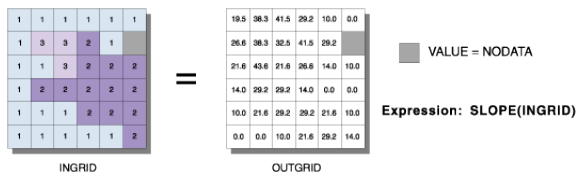
Observer Points: Identifies exactly which observer points are visible from each surface location within the raster dataset.

`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.



Terrain toolset

Contains tools used to create and manage terrain datasets.

Add Feature Class to Terrain: Adds one or more feature classes to a terrain dataset.

`AddFeatureClassToTerrain <in_terrain> <in_feature_class> {height_field} {SF_type} {group} {min_resolution} {max_resolution} {overview} {embedded} {embedded_name} {embedded_fields}; <in_feature_class> {height_field} {SF_type} {group} {min_resolution} {max_resolution} {overview} {embedded} {embedded_name} {embedded_fields}...>`

- The feature classes must reside in the same feature dataset as the terrain.

- Adding a new feature class to a terrain invalidates it. Use the Build Terrain tool after adding feature classes.
- The input terrain cannot be registered as versioned.
- The input feature class cannot be registered as versioned.

Add Terrain Points: Adds multipoints to an embedded terrain feature class by appending them or by using them to replace existing multipoints that fall within the same extent.

`AddTerrainPoints <in_terrain> <terrain_feature_class> <in_feature_class>
{APPEND | REPLACE}`

- This tool functions only on feature classes embedded in a terrain. The terrain and the embedded feature class must already exist.
- If you want to add points to a regular feature class potentially referenced by a terrain but not embedded in it, use the Append tool.
- Adding points to an embedded feature class will invalidate the terrain. Run Build Terrain after adding points.

Add Terrain Pyramid Level: Adds a new pyramid level to an existing terrain dataset.

`AddTerrainPyramidLevel <in_terrain> <ZTOLERANCE> <pyramid_level_definition;pyramid_
level_definition...>`

- Adding a new pyramid level to a terrain invalidates it. Use the Build Terrain tool after adding a pyramid level.
- The input terrain cannot be registered as versioned.

Build Terrain: Performs the necessary steps to make a terrain usable after it's initially defined.

`BuildTerrain <in_terrain>`

- Will perform the work necessary to update the terrain relative to any edits. These include modifications to terrain schema (for example, feature class participation, pyramid definition) or to geometry in associated feature classes. It's more efficient to make a collection of edits followed by one build rather than running a build after each individual edit.

Change Terrain Reference Scale: Changes the reference scale used by a terrain pyramid level.

`ChangeTerrainReferenceScale <in_terrain> <old_refscale> <new_refscale>`

- This tool is useful when an existing reference scale associated with a terrain pyramid level is known to be inappropriate and needs to be reset.
- To add or remove a pyramid level, rather than change the definition of an existing pyramid level, use the Add Terrain Pyramid or Remove Terrain Pyramid tools.

Create Terrain: Creates a new terrain dataset inside the specified feature dataset.

`CreateTerrain <in_feature_dataset> <out_terrain_name> <average_point_spacing> {max_
overview_size} {config_keyword} {WINDOWSIZE | ZTOLERANCE} {ZMIN | ZMAX | ZMEAN | ZMINMAX}
{NONE | MILD | MODERATE | STRONG} {secondary_thinning_threshold}`

- This tool is geared toward data automation for use in scripts and ModelBuilder. Consider using the Terrain wizard in ArcCatalog for interactively creating a new terrain.
- The Average Point Spacing parameter needs to be determined by the data that will be used to build the terrain. This value doesn't need to be exact but should represent a good approximation. If the data has been gathered at significantly different densities from one location to another, give more weight to the smaller spacing.

Remove Feature Class from Terrain: Removes reference to a feature class participating in a terrain dataset.

`RemoveFeatureClassFromTerrain <in_terrain> <feature_class>`

- This tool does not delete the feature class. Rather, it eliminates the terrain dataset's reference to it.
- If the feature class is embedded in the terrain, it will be extracted and dereferenced.
- The terrain will need to be rebuilt using Build Terrain after using this tool.

Remove Terrain Points: Removes points within an area of interest from one or more embedded feature classes.

`RemoveTerrainPoints <in_terrain> <data_source;data_source...> <aoi_extents>`

- This tool functions only on feature classes embedded in a terrain. The terrain and at least one embedded feature class must already exist.
- A rectangular area of interest (AOI) is required. This defines the area in the embedded feature class where multipoint vertices will be deleted. If a multipoint crosses the AOI boundary, only those vertices within the boundary will be deleted.
- Remove Terrain Points looks for an existing edit session to piggy-back on. This supports undo if the edit session was initialized to support undo (for example, using the Editor inside ArcMap). If there's no edit session, it starts and stops one itself, in which case there is no chance for an undo.
- Removing points from an embedded feature class will invalidate the terrain. Run Build Terrain after adding points.

Remove Terrain Pyramid Level: Removes a pyramid level from a terrain dataset.

`RemoveTerrainPyramidLevel <in_terrain> <resolution>`

- You have the option of removing any but the highest resolution pyramid (level 0).
- When used in an ArcSDE database, the input terrain cannot be registered as versioned.



TIN Creation toolset

Contains tools to create and edit TINs.

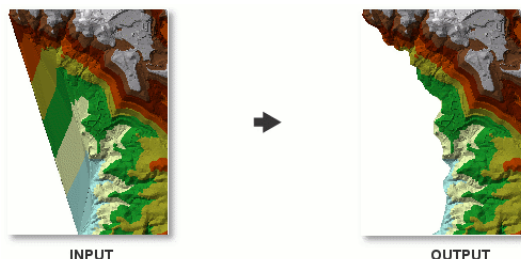
Create TIN: Creates an empty TIN with an extent based on an input geographic dataset.

`CreateTin <out_tin> <spatial_reference>`

- The output TIN cannot already exist.
- Use the Import button on the spatial reference dialog box to browse for the region of the TIN you are creating, 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.

Delineate TIN Data Area: Defines the data area or interpolation zone of a TIN based on triangle edge length.

`DelineateTinDataArea <in_tin> <max_edge_length> {PERIMETER_ONLY | ALL}`



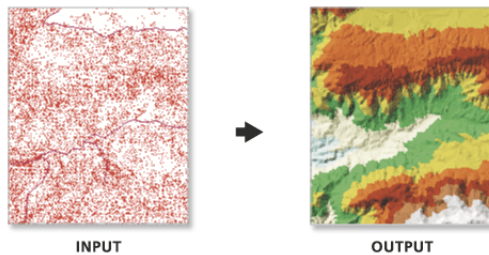
INPUT

OUTPUT

- This tool modifies the input TIN. If you would prefer not to modify the input, use the standard Copy tool and operate on the copied TIN.
- The value for <max_edge_length> is best determined from the average spacing of nodes in the TIN within areas that are considered to be valid data zones. Provide a value that is larger than the average spacing.

Edit TIN: Adds feature classes to an existing TIN and creates surface features of a TIN based on an input feature class.

```
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}...>
```



- 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 Create TIN tool.
- Features may be added to a newly created or existing TIN.

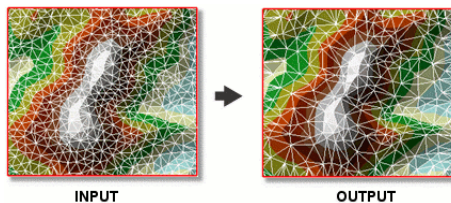


TIN Surface toolset

Contains tools to analyze the surface of a TIN dataset.

Decimate TIN Nodes: Produces a TIN that is a generalized version of another.

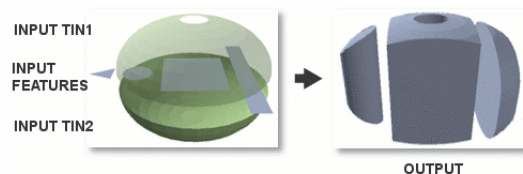
```
DecimateTinNodes <in_tin> <out_tin> <method> {NO_BREAKLINES | BREAKLINES}
```



- A subset of nodes from an input TIN is selected to produce the new TIN.

Extrude Between: Converts polygons into multipatches by extruding them between two input TINs.

```
ExtrudeBetween <in_tin1> <in_tin2> <in_feature_class> <out_feature_class>
```



- The output multipatches are written to a new feature class.
- The input TINs and polygons must all overlap in horizontal extent to produce any output.

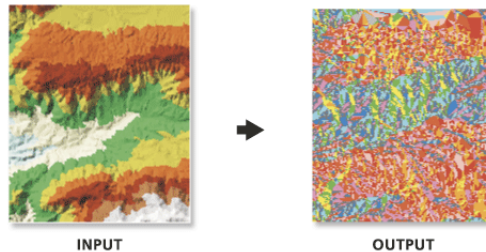
Interpolate Polygon to Multipatch: Creates surface-conforming areal features by extracting those portions of a surface that fall within the extent of input polygons as multipatches.

```
InterpolatePolyToPatch <in_tin> <in_feature_class> <out_feature_class> {max_strip_size}  
    {z_factor} {area_field} {surface_area_field}
```

- The attributes from the input features are copied to the output. Planimetric and surface area is calculated for each feature and added as attribution to the output.
- Convert polygons to multipatches if you are having display problems with polygons conforming to a surface.
- The max_strip_size value must be three or larger. The recommended range is between 128 and 2048.

TIN Aspect: Extracts aspect information from an input TIN into an output feature class.

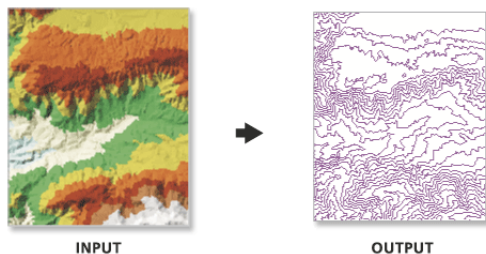
`TinAspect <in_tin> <out_feature_class> {class_breaks_table} {aspect_field}`



- Aspect is expressed in degrees.

TIN Contour: Creates a feature class containing a set of contours generated from a functional surface.

`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 Difference: Calculates the volumetric difference between two TINs.

`TinDifference <in_tin1> <in_tin2> <out_feature_class>`

- The input TINs need to overlap in horizontal extent.
- It's best if the horizontal and vertical coordinate systems of the input TINs are the same.

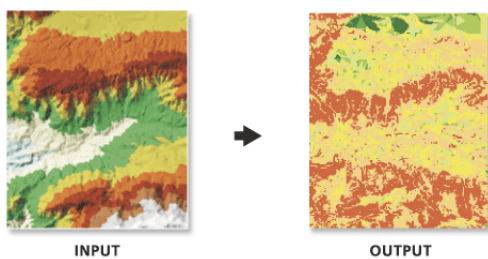
TIN Polygon Volume: Calculates the volumetric and surface area between polygons of an input feature class and a TIN surface.

`TinPolygonVolume <in_tin> <in_feature_class> <in_height_field> {BELOW | ABOVE}
{out_volume_field} {surface_area_field}`

- The input polygons and TIN need to overlap in horizontal extent for surface area or volume to be calculated.

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 the `class_breaks_table` parameter 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 into 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 into 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.

Calculate Z-Value: Uses the interpolation model in a geostatistical layer to predict a value at a single location.

`GACalculateZValue <in_geostat_layer> <point_coord>`

- This tool is generally used in a model or in scripting.

Create Geostatistical Layer: Uses an existing geostatistical layer to create a new layer, which includes a new feature dataset or variable.

`GACreateGeostatisticalLayer <in_ga_model_source> <in_datasets> <out_layer>`

- Geostatistical model source is either a geostatistical layer or a geostatistical model (XML).
- The dataset used to create the geostatistical layer is persisted in the layer. However, when an XML version is saved, only the model parameters are saved and not the dataset information.

GA Layer to Contour: Exports a geostatistical layer to contours.

`GALayerToContour <in_geostat_layer> <CONTOUR | FILLED CONTOUR> <out_feature_class> {Draft | Presentation}`

- You can choose to display the contours in either draft or presentation quality.
- CONTOUR is the contour or isoline representation of the geostatistical layer. You can display the lines in either draft or presentation quality.
- FILLED_CONTOUR is the polygon representation of the geostatistical layer. It is assumed for the graphical display that the values between the polygons are the same for all locations within the polygon. You can display the lines in either draft or presentation quality.

GA Layer to Grid: Exports a geostatistical layer to a grid.

`GALayerToGrid <in_geostat_layer> <output_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.

GA Layer to Points: Exports a geostatistical layer to points.

`GALayerToPoints <in_geostat_layer> <in_locations> {z_field} <out_feature_class>`

- If a field is left blank, predictions will be made at the location points.
- If a field is selected, predictions will be made and compared to the values in the Z_value_field and a validation analysis is performed.

Gaussian Geostatistical Simulations: Performs conditional or unconditional geostatistical simulation based on a Simple Kriging model.

`GaussianGeostatisticalSimulations <in_geostat_layer> <number_of_realizations> <output_workspace> <output_simulation_prefix> {in_conditioning_features} {conditioning_field} {cell_size} {in_bounding_dataset} {save_simulated_rasters} {quantile} {threshold} {in_stats_polygons} {MIN | MAX | MEAN | STDDEV | QUARTILE1 | MEDIAN | QUARTILE3 | QUANTILE | P_THRSHLD}`

- The geostatistical layer used as input must be the result of performing Simple Kriging on a dataset. Geostatistical layers resulting from other types of kriging should not be used with this tool. Additionally:
 - For point data at 1:100,000 scale, use 5.0 and 200.0.

- A Normal Score transform of the data is necessary to guarantee that the input data follows a standard normal distribution.
- Clustered data should be declustered (using either the cell or polygon with a clipping outline method) so that the input histogram accurately represents the sampled population. This histogram will be reproduced in the realizations.
- To generate conditional realizations, the conditioning data should be the same as the data that was used to construct the Simple Kriging model that the simulation will be based on. However, other datasets can be used to condition the realizations.
- Output generated can be identified as follows:
 - The prefix followed by s0 to sN (where N is the number of realizations) is used to name the simulated rasters when the option save_simulated_rasters is used.
 - The prefix followed by MIN, MAX, MEAN, STDDEV, QUARTILE1, MEDIAN, QUARTILE3, QUANTILE or P_THRSHLD is used to name the output rasters when these postprocessing options have been selected.
 - The prefix followed by the polygon feature class name is used to name the output polygon feature class when postprocessing for areas of interest (statistical polygons) has been selected.

Get Model Parameter: Gets model parameter value from an existing geostatistical model source.

`GAGetModelParameter <in_ga_model_source> <model_param_xpath>`

- This tool is generally used in a model or in scripting.
- The geostatistical model source is either a geostatistical layer or a geostatistical model (XML).

Moving Window Kriging: Automatically estimates the kriging model for each neighborhood as the kriging interpolation moves through all the location points.

`GAMovingWindowKriging <in_ga_model_source> <in_datasets> <in_locations> <neighbors_max> <out_feature_class> {cell_size} {out_surface_grid}`

- The geostatistical model source is either a geostatistical layer or a geostatistical model (XML).
- The input dataset must have more than 10 points. However, it is recommended that it has more than 300 points.

Neighborhood Selection: Creates a layer of points based on a user-defined neighborhood.

`GANeighborhoodSelection <in_dataset> <out_layer> <point_coord> <neighbors_max> <neighbors_min> <minor_semiaxis> <major_semiaxis> <angle> {ONE SECTOR | FOUR SECTORS | FOUR SECTORS SHIFTED | EIGHT SECTORS}`

Semivariogram Sensitivity: Varies the semivariogram parameters Nugget, Partial Sill, and Range to perform a sensitivity analysis.

`GAsemivariogramSensitivity <in_ga_model_source> <in_datasets> <in_locations> {nugget_span_percents} {nugget_calc_times} {partialsill_span_percents} {partialsill_calc_times} {range_span_percents} {range_calc_times} {minrange_span_percents} {minrange_calc_times} <out_table>`

- The geostatistical model source is either a geostatistical layer or a geostatistical model (XML).

Set Model Parameter: Sets parameter value in an existing geostatistical model source.

`GASetModelParameter <in_ga_model_source> <model_param_xpath> <in_param_value> <out_ga_model>`

- This tool is generally used in a model or in scripting.
- The geostatistical model source is either a geostatistical layer or a geostatistical model (XML).

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 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 tools 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} {NO_SNAP | SNAP} {snap_offset}
```

- 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, 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 | NAUTICALMILES> {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 | NO_LINES | STRAIGHT_LINES | TRUE_LINES_WITHOUT_MEASURES}
```

- 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 only be set if the input analysis network has a hierarchy attribute.

Make OD Cost Matrix Layer: Makes an origin and destination (OD) 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_LINES}
```

- 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 only be set 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} {NO_TIMEWINDOWS | USE_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 | NO_LINES | STRAIGHT_LINES |
  TRUE_LINES_WITHOUT_MEASURES} {start_date_time}
```

- 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 only be set 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_POLYS | DETAILED_
POLYS | NO_POLYS} {NO_MERGE | NO_OVERLAP | MERGE} {RINGS | DISKS} {NO_LINES | TRUE_LINES |
TRUE_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...} {TRIM_POLYS | NO_TRIM_POLYS} {poly_trim_value}
{NO_LINES_SOURCE_FIELDS | LINES_SOURCE_FIELDS}
```

- 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 only be set if the input analysis network has a hierarchy attribute.

Make Vehicle Route Problem Layer: Creates a vehicle routing problem (VRP) network analysis layer and sets its analysis properties.

```
MakeVehicleRoutingProblemLayer <in_network_dataset> <out_network_analysis_layer>
  <time_impedance> {distance_impedance} {Minutes | Seconds | Hours | Days} {MILES |
  KILOMETERS | FEET | YARDS | METERS | INCHES | CENTIMETERS | MILLIMETERS | DECIMETERS |
  NAUTICALMILES | DECIMALDEGREES} {default_date} {capacity_count} {MEDIUM | HIGH | LOW}
  {MEDIUM | HIGH | LOW} {ALLOW_UTURNS | NO_UTURNS | ALLOW_DEAD_ENDS_ONLY} {restriction_
  attribute_name; restriction_attribute_name...} {NO_HIERARCHY | USE_HIERARCHY}
  {hierarchy_settings} {TRUE_LINES_WITH_MEASURES | TRUE_LINES_WITHOUT_MEASURES |
  STRAIGHT_LINES | NO_LINES}
```

- A vehicle routing problem analysis layer can be used to solve common fleet management problems such as servicing a set of orders using a fleet of vehicles.
- The vehicle routing problem layer can be made on a network dataset currently added to the ArcMap table of contents or on a network dataset on disk.

- Hierarchy options can only be set 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 <in_network_analysis_layer> {SKIP | HALT}`

- Be sure to specify all the parameters necessary to perform the analysis before running Solve.



Network Dataset toolset

Contains a tool used to build network datasets.

Build Network: Reconstructs the network connectivity and attribute information of a network dataset.

`BuildNetwork <in_network_dataset>`

- You can build the network without starting ArcCatalog after making edits to the attributes or the features of a participating source feature class.



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>`

- Once the maximum number of edges is increased, it cannot be decreased later. So only increase by the needed amount.
- Increasing the number of edges by one adds three additional fields to the turn feature class. Take care not to exceed the maximum number of fields allowed by the database being used. For example, personal geodatabase is limited to 255 fields.

Populate Alternate ID Fields: Creates and populates additional fields on the turn feature class(es) in a network dataset that reference the edges by alternate IDs.

`PopulateAlternateIDFields <in_network_dataset> <alternate_ID_field_name>`

- The alternate IDs allow for another set of IDs that can help maintain the integrity of the turn features in case the source edges are being edited.

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}`

- The turn feature class to be created is placed in the same location as the reference line feature class.

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>`

- Use Populate Alternate ID Fields tool before making edits to the input line features. Then this tool can be used to synchronize the turn features based on the alternate ID fields.

Update by Geometry: Updates all the edge references in the turn table using the geometry of the feature.

`UpdateByGeometry <in_turn_features>`

- This tool is useful when the IDs listed for the turn can no longer find the edges participating in the turn.

Schematics toolbox

Contains tools used to create, update, export diagrams, or create schematic folders when using the Schematics extension.

ConvertDiagram: Converts a schematic diagram to feature classes or shapefiles.

`ConvertDiagram <in_diagram> <out_location> {REUSE_FC | NO_REUSE_FC} {NO_RELATED_ATTRIBUTES | EXPORT_RELATED_ATTRIBUTES} {POLYGON | POLYLINE | POINT} {config_keyword}`

- To convert a schematic diagram to feature classes, define a geodatabase or feature dataset for the specified out_location. To convert a schematic diagram to shapefiles, define a folder.
- To export several diagrams in the same feature classes or shapefiles, specify REUSE_FC. To export several diagrams in different feature classes or shapefiles, specify NO_REUSE_FC.

CreateDiagram: Creates a schematic diagram.

`CreateDiagram <out_location> <out_name> <diagram_type> {in_data;in_data...} {builder_options}`

- Depending on the schematic builder, the diagram creation can be based on feature layers, feature classes, object tables, a solved network analysis, or a XML file.
- If the output schematic diagram name already exists, it may be deleted before being recreated. To avoid this deletion, you can uncheck the Overwrite outputs of geoprocessing operations option from the Geoprocessing tab in the Options dialog.

CreateSchematicFolder: Creates a schematic folder in a schematic dataset or schematic folder.

`CreateSchematicFolder <out_location> <out_name>`

- The output schematic folder may already exist but in this case, it will be not recreated even if you are working with the Overwrite the outputs of geoprocessing operations option checked.

UpdateDiagram: Updates a schematic diagram.

`UpdateDiagram <in_diagram> {in_data;in_data...} {builder_options}`

- Depending on the schematic builder, the diagram update can be based on feature layers, feature classes, object tables, a solved network analysis, or a XML file.
- The in_diagram parameter must specify the exact location for the diagram to be updated.
- If a particular layout was saved for the specified schematic diagram, Schematics synchronizes the display of all the updated schematic elements with the saved layout. Schematic elements that were in the diagram before update are displayed according to their last saved position, and new elements that may have been introduced during the update are positioned at their geographical coordinates.

UpdateDiagrams: Updates schematic diagrams stored in a schematic dataset or schematic folder.

`UpdateDiagrams <in_container> {builder_options} {RECURSIVE | NO_RECURSIVE} {diagram_type} {last_update_criteria}`

- All diagrams or a subset of diagrams (for example, diagram related to a specific diagram type, or diagrams that have not been updated for a particular number of days) can be updated.
- Only diagrams based on the Custom Query Based and Standard builder diagrams can be updated using this geoprocessing tool.
- Diagrams based on the Network Dataset and XML builders that require specific input data cannot be updated using this tool.

- For diagrams based on the Standard builder, the update operates from the initial feature set used to initially generate the specified schematic diagram or from the updated trace result based on the trace parameters that were persisted in the schematic database if the diagram was generated from an highlighted trace.

Spatial Analyst toolbox

This toolbox provides tools for performing cell-based (raster) spatial analysis.

Conditional toolset

Contains tools to control the output values based on the conditions placed on the input values.

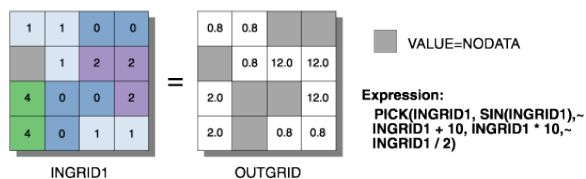
Con: Performs a conditional if/else evaluation on each of the input cells of an input raster.

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.

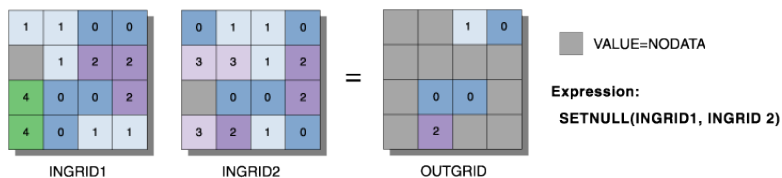
Pick <in_position_raster> <in_rasters_or_constants; in_rasters_or_constants...>
<out_raster>



- 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 zero 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 zero.



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.

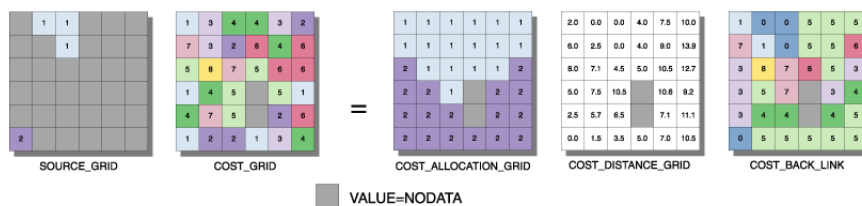
Corridor: Calculates the sum of accumulative costs for two input accumulative cost rasters.

`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: Calculates for each cell its nearest source based on the least accumulative cost over a cost surface.

`CostAllocation <in_source_data> <in_cost_raster> <out_allocation_raster> {maximum_distance} {in_value_raster} {source_field} {out_distance_raster} {out_backlink_raster}`

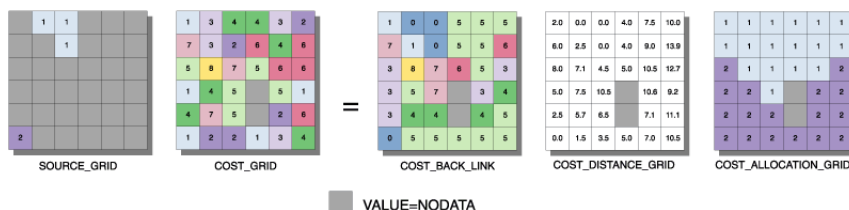


Expression: `COSTALLOCATION(SOURCE_GRID,COST_GRID,COST_DISTANCE_GRID,COST_BACK_LINK)`

- 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 zero is considered a legitimate source. A source raster may 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; either 1 or zero. 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 that is the next cell on the least accumulative cost path to the nearest source.

`CostBackLink <in_source_data> <in_cost_raster> <out_backlink_raster> {maximum_distance} {out_distance_raster}`



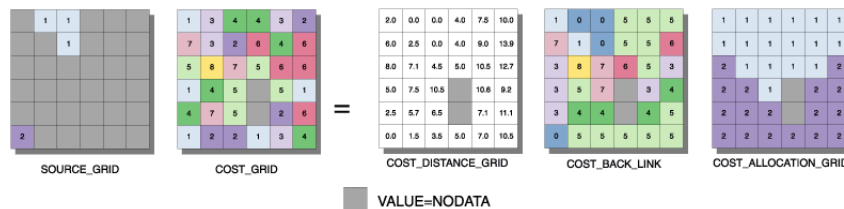
Expression: `COSTBACKLINK(SOURCE_GRID,COST_GRID,COST_DISTANCE_GRID,COST_ALLOCATION_GRID)`

- 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 zero is considered a legitimate source. A source raster may 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 zero. 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 for each cell to the nearest source over a cost surface.

`CostDistance <in_source_data> <in_cost_raster> <out_distance_raster> {maximum_distance} {out_backlink_raster}`



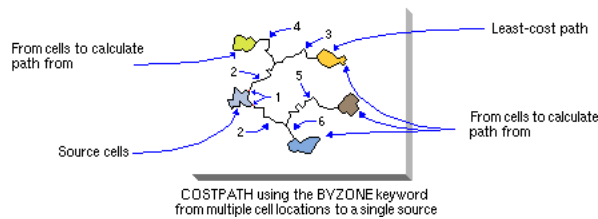
Expression: `COSTDISTANCE(SOURCE_GRID,COST_GRID,
COST_BACK_LINK,COST_ALLOCATION_GRID)`

- 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 zero is considered a legitimate source. A source raster may 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}`

- Cost path 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 zero 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.



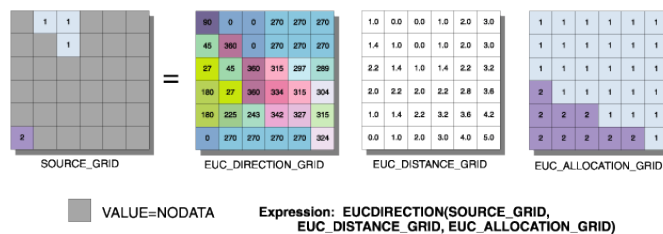
Euclidean Allocation: Calculates, for each cell, the nearest source based on Euclidean distance.

`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 zero 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 zero. 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.

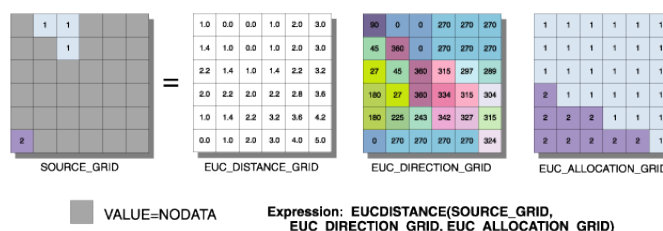
`EucDirection <in_source_data> <out_direction_raster> {maximum_distance} {cell_size} {out_distance_raster}`



- 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 zero 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 zero 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 Euclidean distance 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 zero 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 to the nearest source, 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 zero 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 the nearest source for each cell based on the least accumulative cost over a cost surface, while accounting for surface distance and horizontal 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 zero 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 zero 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.

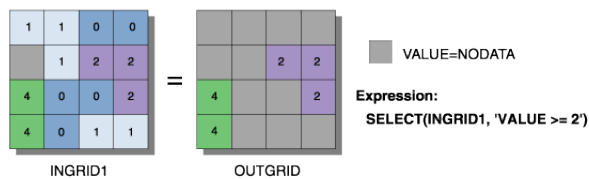


Extraction toolset

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.

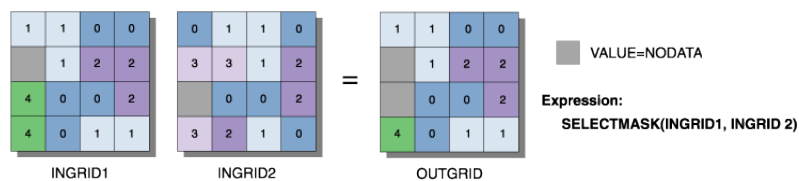
Extract by Circle: Extracts the cell values of a raster dataset based on the boundaries of a circle.

`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 cells of a raster dataset that correspond to the areas defined by a mask.

`ExtractByMask <in_raster> <in_mask_data> <out_raster>`



- Where the input raster or feature mask data is a 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 tools.
- 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 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: Creates a table that shows the values of cells from a raster, or set of rasters, for defined locations.

`Sample <in_rasters;in_rasters...> <in_location_data> <out_table> {NEAREST | BILINEAR | CUBIC}`

- The locations are defined by raster cells or by a set of points.
- 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 zero 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.

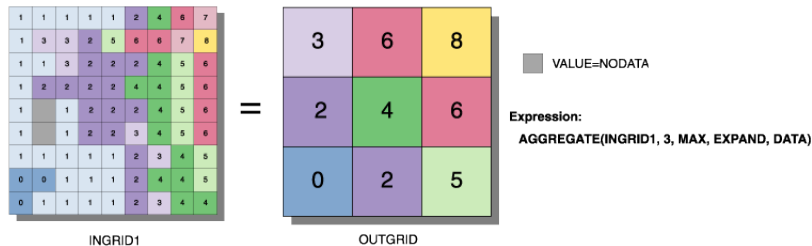


Generalization toolset

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.

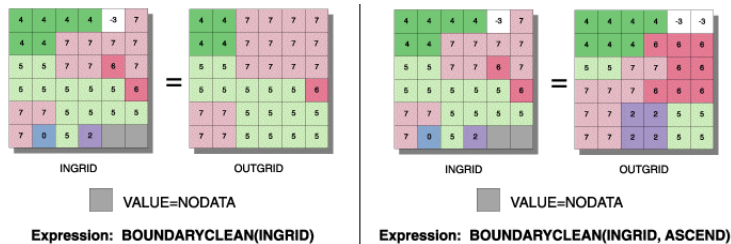
Aggregate <in_raster> <out_raster> <cell_factor> {SUM | MAXIMUM | MEAN | MEDIAN | MINIMUM}
{EXPAND | TRUNCATE} {DATA | NODATA}



- 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.

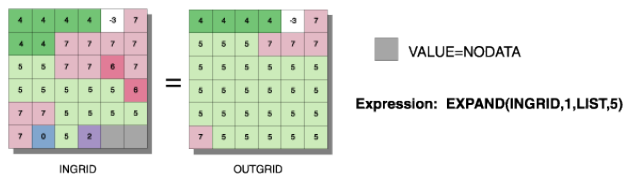
BoundaryClean <in_raster> <out_raster> {NO_SORT | DESCEND | ASCEND} {TWO_WAY | ONE_WAY}



- 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.

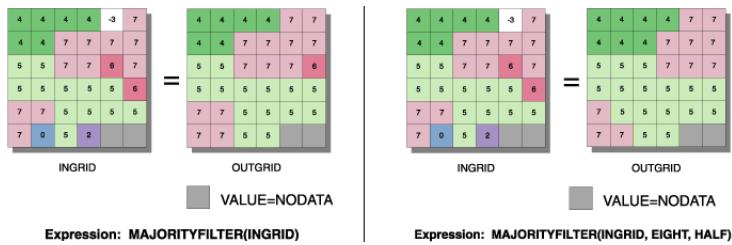
Expand <in_raster> <out_raster> <number_cells> <zone_values;zone_values...>



- 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.

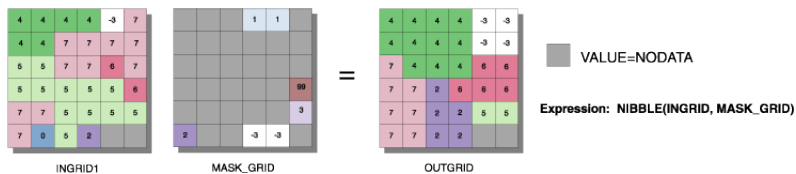
MajorityFilter <in_raster> <out_raster> {FOUR | EIGHT} {MAJORITY | HALF}



- 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.
- While 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.

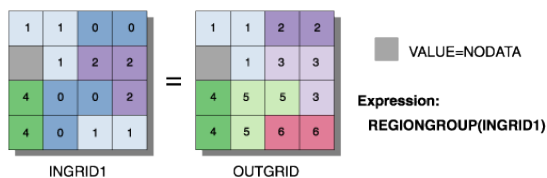
`nibble <in_raster> <in_mask_raster> <out_raster> {ALL_VALUES | DATA_ONLY}`



- 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.

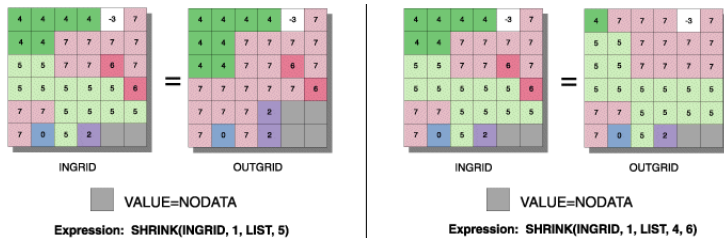
`RegionGroup <in_raster> <out_raster> {FOUR | EIGHT} {WITHIN | CROSS} {ADD_LINK | NO_LINK} {excluded_value}`



- 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 by reducing the number of cells representing the width of the features.

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 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 zero and 1, with typical values around 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 zero and 1, with typical values around 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, returning an ASCII file of particle tracking data and, optionally, a coverage of track information.

```
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 time-dependent, two-dimensional concentration distribution in mass per volume of a solute introduced instantaneously and at a discrete point into a vertically mixed aquifer.

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 around 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.

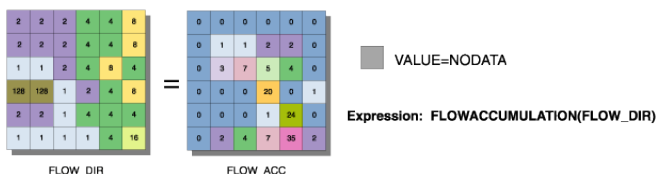
Fill: Fills sinks in a surface raster to remove small imperfections in the data.

`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.

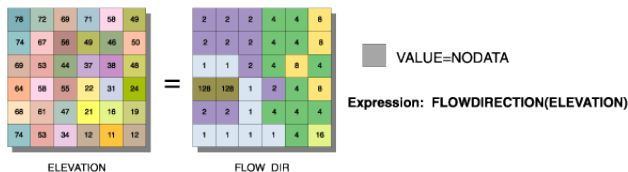
`FlowAccumulation <in_flow_direction_raster> <out_accumulation_raster> {in_weight_raster}`
{FLOAT | INTEGER}



- 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 the steepest drop was 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 distance or weighted distance along a flow path.

`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.

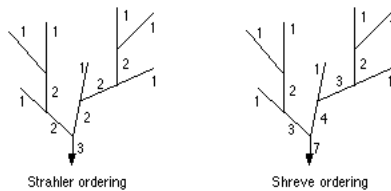
`StreamLink <in_stream_raster> <in_flow_direction_raster> <out_raster>`

- 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 one 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 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).

Kriging: Interpolates a raster dataset from a set of points using kriging.

```
Kriging <in_point_features> <z_field> <out_surface_raster> <semiVarioqram_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 have to 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.

 **Spline with Barriers:** Interpolates a surface, using barriers, from points using a minimum curvature spline technique. The barriers are entered as either polygon or polyline features.

```
SplineWithBarriers <input_point_features> <z_value_field> {input_barrier_features} <output_cell_size> <output_raster> {smoothing_factor}
```

Topo to Raster: Interpolates a hydrologically correct surface from point, line, and polygon data.

```
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 $nrows \times ncols$, 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: Interpolates a hydrologically correct surface from point, line, and polygon data using parameters specified in a file.

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 -810480.6250000000 8321785.0000000000 810480.6250000000
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}
{out_rms_file}

- 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 LOGISTIC option of Type of Regression, the Z value field of input point features should have codes of zero 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.

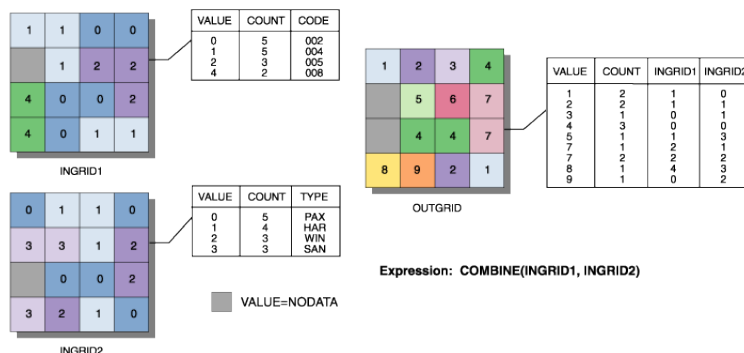
Cell Statistics: Calculates a per-cell statistic from multiple raster datasets.

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 rasters so a unique output value is assigned to each unique combination of input values.

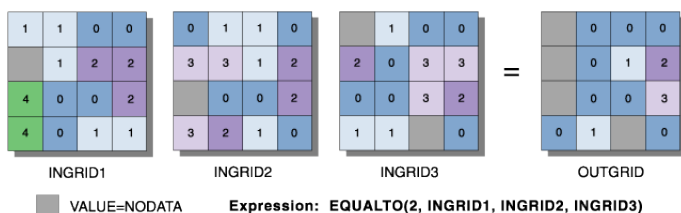
combine <in_rasters;in_rasters...> <out_raster>



- 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.

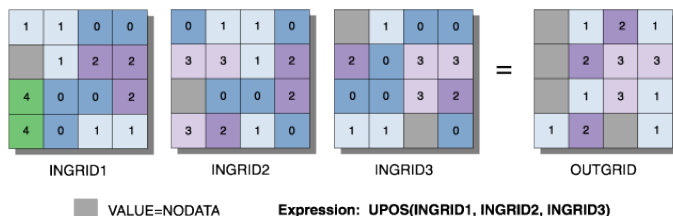
`GreaterThanFrequency <in_value_raster> <in_rasters;in_rasters...> <out_raster>`



- 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.

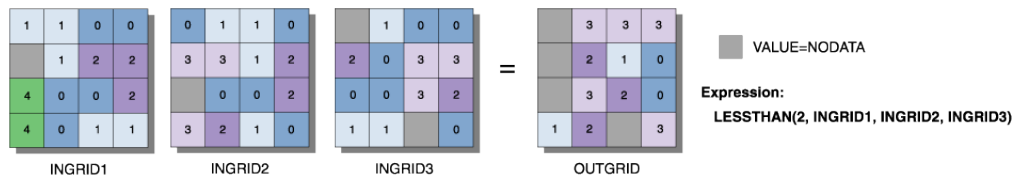
`HighestPosition <in_rasters_or_constants;in_rasters_or_constants...> <out_raster>`



- 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.

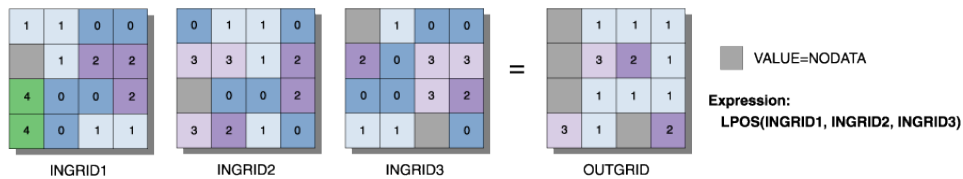
`LessThanFrequency <in_value_raster> <in_rasters;in_rasters...> <out_raster>`



- An arbitrary number of rasters can be specified in the input rasters list.

Lowest Position: Determines the position of a raster with the minimum value in a set of rasters.

`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 rasters based on a rank level specified by another raster.

`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: Runs an expression built with the Map Algebra language.

`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 grid 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: Runs a single expression built with the Map Algebra language.

`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.

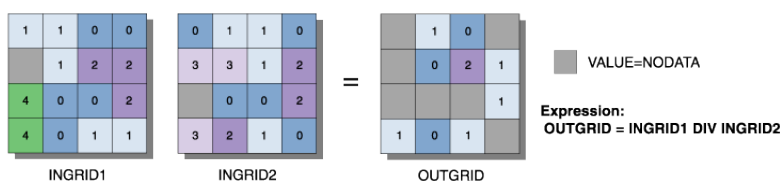
`Abs <in_raster_or_constant> <out_raster>`



- 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>`

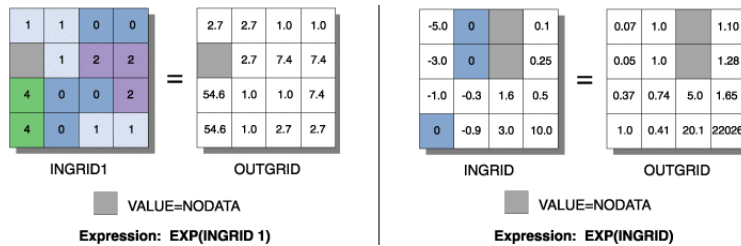


- The order of input is relevant for Divide.
- When a number is divided by zero, the output result is NoData.

- 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.

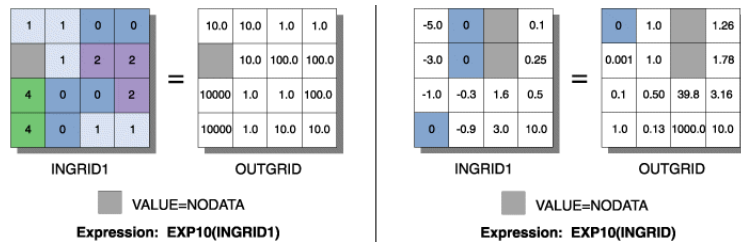
Exp <in_raster_or_constant> <out_raster>



- The base e exponential is the most commonly used exponential function.

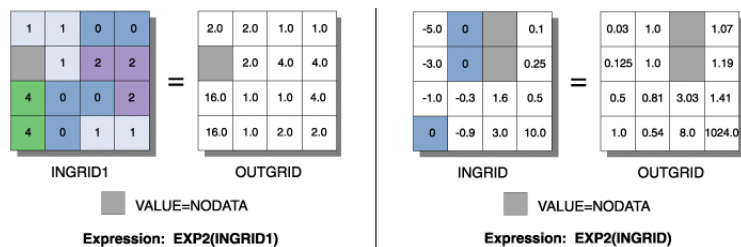
Exp10: Calculates the base 10 exponential of cells in a raster dataset.

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.

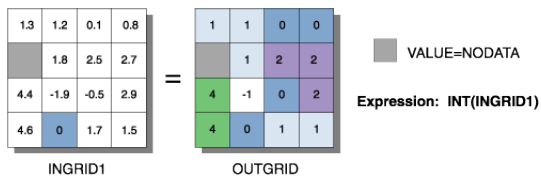
Float <in_raster_or_constant> <out_raster>



- 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.

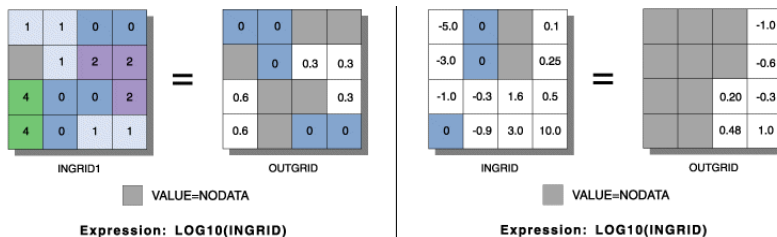
`Ln <in_raster_or_constant> <out_raster>`



- The natural logarithm is the most commonly used logarithmic function.
- The input value of a logarithmic function cannot be zero or a negative number. If it is, the output will be NoData.

Log10: Calculates the base 10 logarithm of cells in a raster dataset.

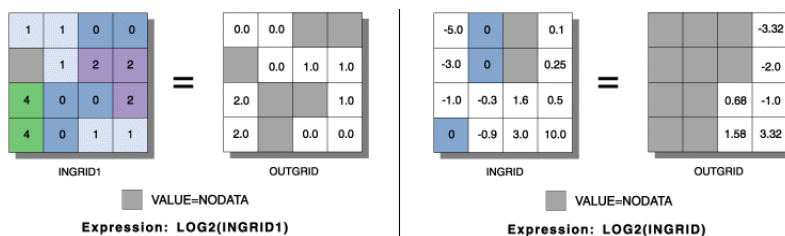
`Log10 <in_raster_or_constant> <out_raster>`



- The input value of a logarithmic function cannot be zero or a negative number. If it is, the output will be NoData.

Log2: Calculates the base 2 logarithm of cells in a raster dataset.

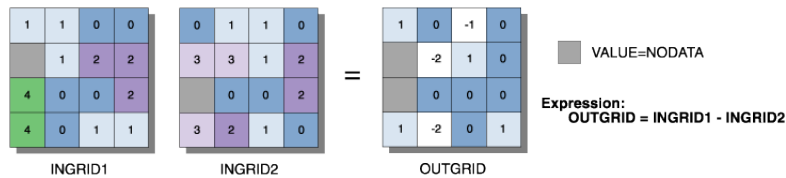
`Log2 <in_raster_or_constant> <out_raster>`



- The input value of a logarithmic function cannot be zero 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.

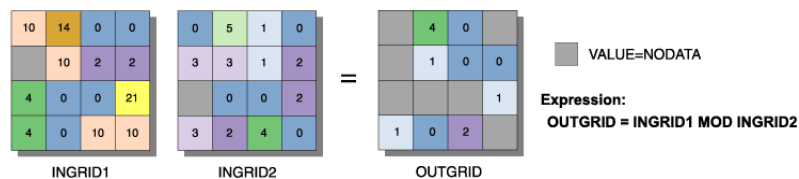
Minus <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- 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 zero is assigned NoData on the output. Therefore, any location on the second input that is either zero 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 integer, those inputs will be converted to integers through truncation. Output values are always integers.

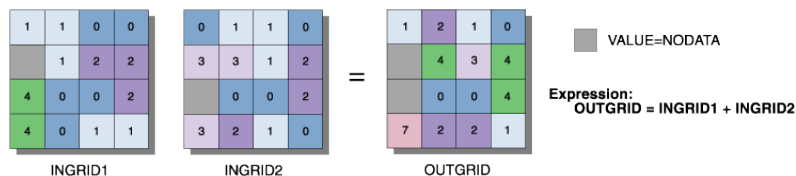
Negate: Changes the sign of the cell values of the input raster dataset (multiplies by -1).

Negate <in_raster_or_constant> <out_raster>



Plus: Adds the values of two raster datasets on a cell-by-cell basis.

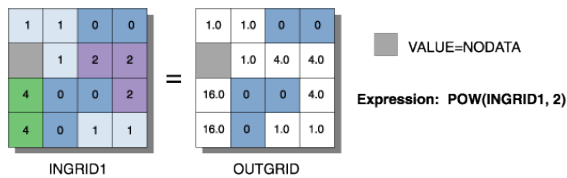
Plus <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- The order of input is irrelevant in the addition expression.

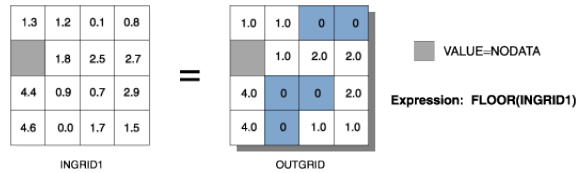
Power: Calculates the nth power of the input raster or number on a cell-by-cell basis.

Power <in_raster_or_constant> <in_raster_or_constants> <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>



- 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:

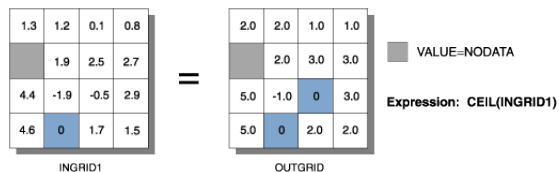
Round Down on 1.5 becomes 1.0

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>

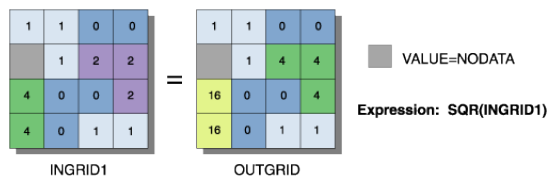


- If a number has any values to the right of the decimal point, the output will be assigned the next highest whole value:

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.

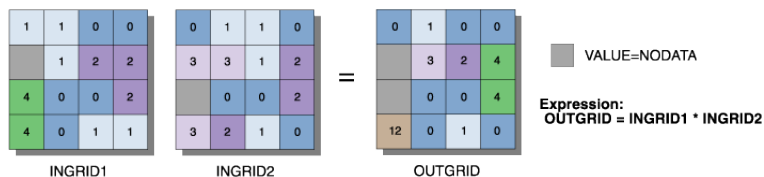
SquareRoot <in_raster_or_constant> <out_raster>



- Input values must be greater than or equal to zero. 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 zero; if negative, the bit is 1.
- Bitwise And 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 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 position is zero; 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 ~ 1010 = 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 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 position is zero; if negative, the bit position is 1.
- The Bitwise Not 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).

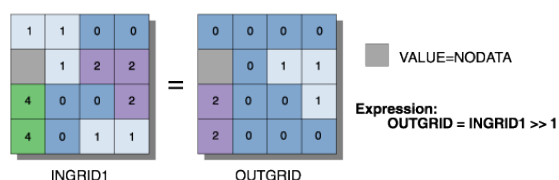
BitwiseOr <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- 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 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 position is zero; 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>

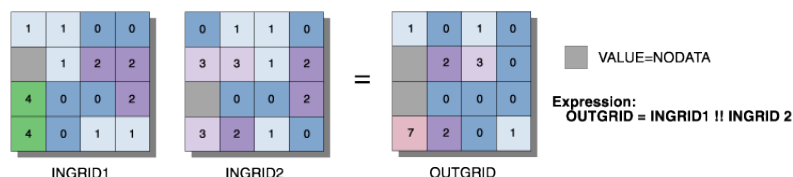


- The bitwise operators work on 32-bit integers.

- 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 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 position is zero; 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 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 position is zero; 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.

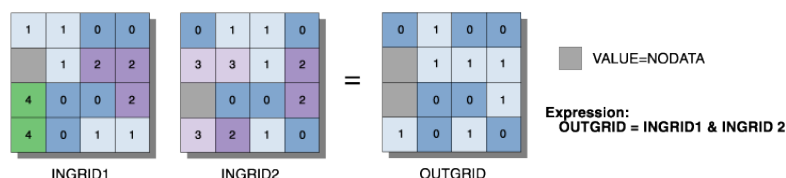


Logical (Math) toolset

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.

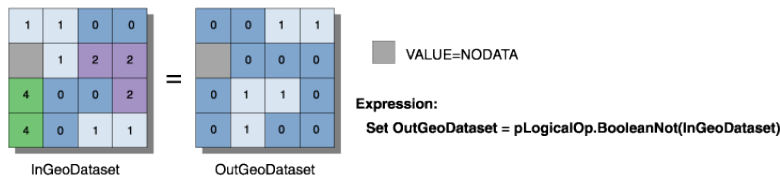
BooleanAnd <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- 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 zero. 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.

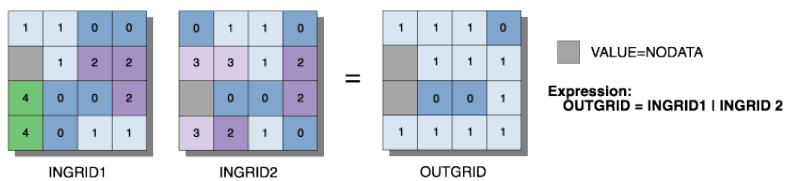
BooleanNot <in_raster_or_constant> <out_raster>



- 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 zero (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.

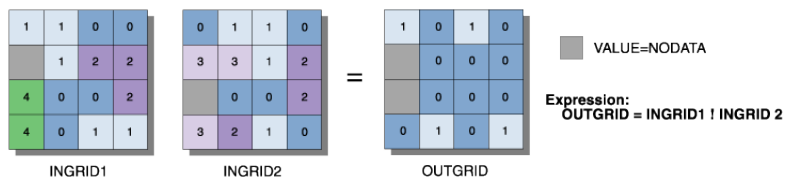
`BooleanOr <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- 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 zero. 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.

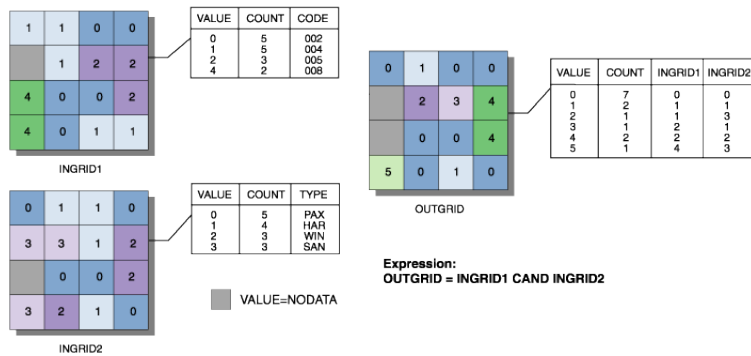
`BooleanXOr <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- 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 zero. If one or both values are NoData, the output value is NoData.

Combinatorial And: Performs a combinatorial AND operation on two input raster datasets.

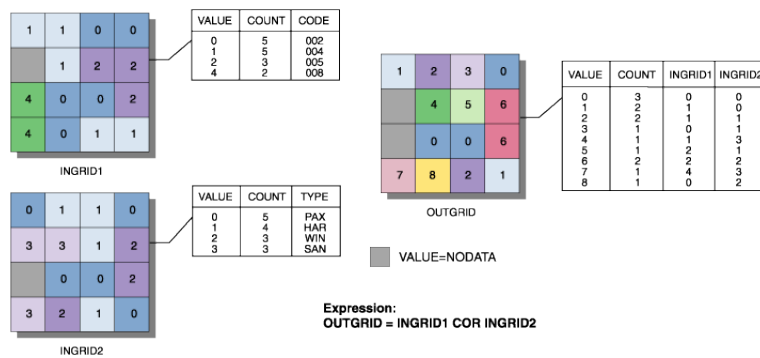
`CombinatorialAnd <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- Both inputs must contain positive integer values.

Combinatorial Or: Performs a combinatorial OR operation on the cell values of two input raster datasets.

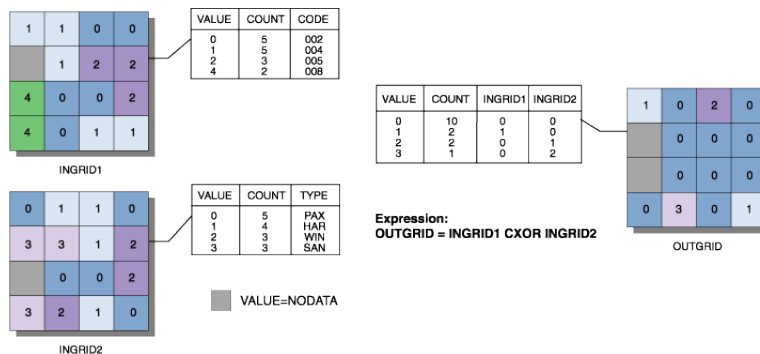
`CombinatorialOr <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- Both inputs must contain positive integer values.

Combinatorial XOR: Performs a combinatorial exclusive OR operation on the cell values of two input raster datasets.

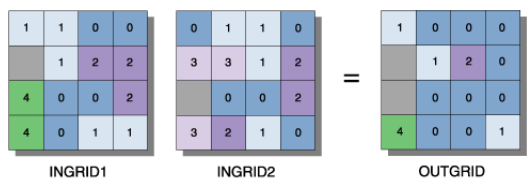
`CombinatorialXor <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- Both inputs must contain positive integer values.

Diff: Determines which values from the first input are logically different from the values of the second input on a cell-by-cell basis.

`Diff <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



Expression:
OUTGRID = INGRID1 DIFF INGRID2

VALUE=NODATA

- If the values on the two inputs are different, the value on the first input is output. If the values on the two inputs are the same, the output is zero.
- The order of input is relevant in the Diff expression.
- If both inputs are integer, the output raster will be integer; otherwise, it will be floating point.
- A number can be used as an input; however, the cell size and extent must first be set in the environment.

Equal To: Returns 1 for cells where the first raster equals the second raster and zero if it does not.

`EqualTo <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`

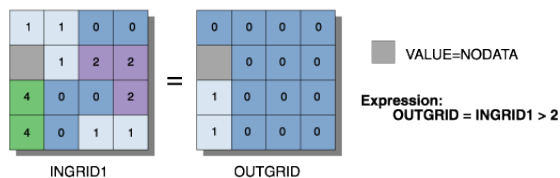


Expression: EQUALTO(2, INGRID1, INGRID2, INGRID3)

- 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 zero. 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 zero where it is not.

`GreaterThan <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`

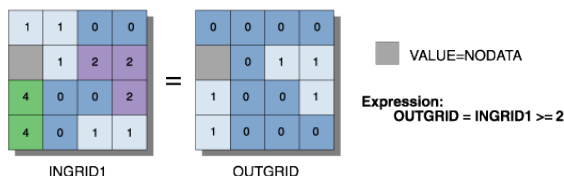


Expression:
OUTGRID = INGRID1 > 2

- 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 zero. 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 zero where it is not.

`GreaterThanEqual <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`

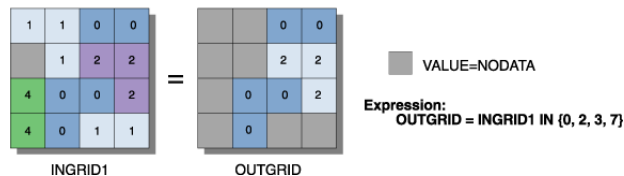


Expression:
OUTGRID = INGRID1 >= 2

- 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 zero. When one or both input values are NoData, the output is NoData.

InList: Determines which values from the first input are contained in the other inputs, on a cell-by-cell basis.

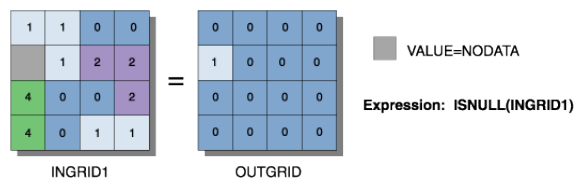
`InList <in_raster_or_constant1> <in_raster_or_constant2;in_raster_or_constant2...> <out_raster>`



- If the first input is a raster and the value of a cell location is not contained in the set specified by the second input, the location will receive NoData on the output raster.
- If all the inputs are integers, the output raster will be integer; otherwise, it will be floating point.
- A number can be used as an input; however, the cell size and extent must first be set in the environment.

Is Null: Returns 1 for cells in the input raster dataset that have a value of NoData and returns zero where they do not.

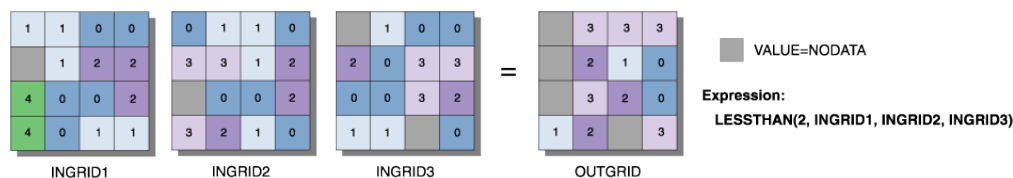
`IsNull <in_raster> <out_raster>`



- The output value type is always integer. The values are either 1 or zero. Cells in the input that have a value are given zero 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 zero where it is not.

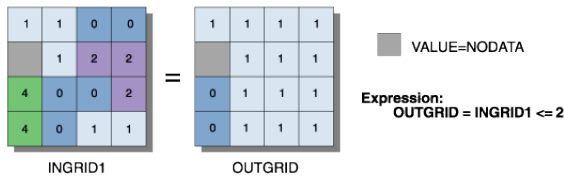
`LessThan <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- 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 zero. 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 zero where it is not.

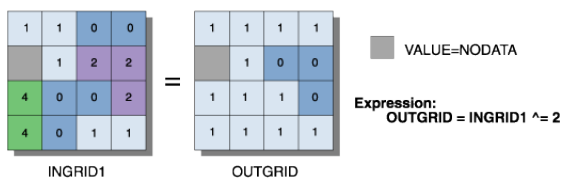
`LessThanEqual <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>`



- 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 zero. 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 zero where it is not.

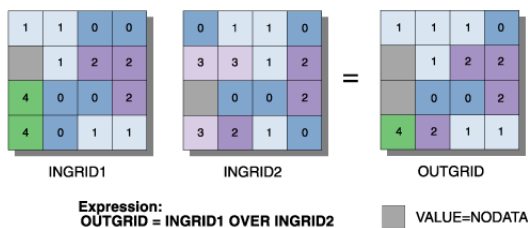
NotEqual <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- 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 zero. When one or both input values are NoData, the relational expression outputs NoData.

Over: Returns those values from the first input that are non-zero; otherwise, returns the value from the second input on a cell-by-cell basis.

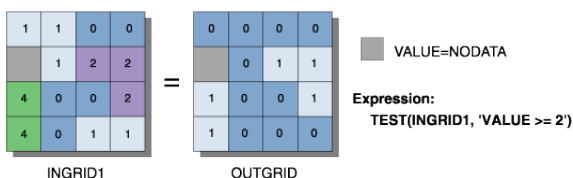
over <in_raster_or_constant1> <in_raster_or_constant2> <out_raster>



- The order of input is relevant in the Over expression.
- If both inputs are integer, the output raster will be integer; otherwise, it will be floating point.
- A number can be used as an input; however, the cell size and extent must first be set in the environment.

Test: Returns 1 for cells that evaluate to true based on a logical expression and returns zero 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 zero (if the test evaluates to false).
- The test is specified by a SQL expression.

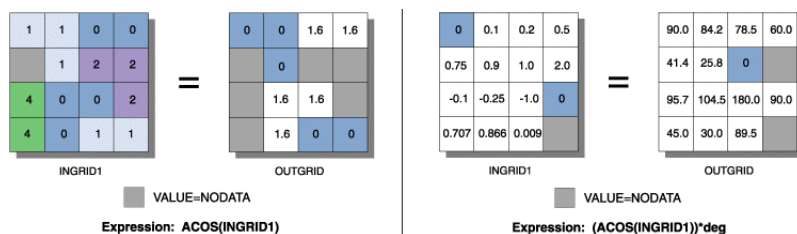


Trigonometric (Math) toolset

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.

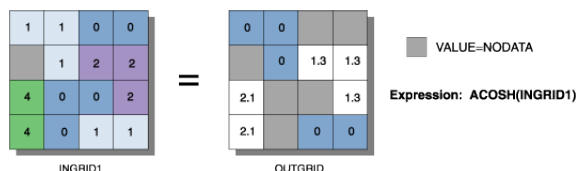
ACos <in_raster_or_constant> <out_raster>



- 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.

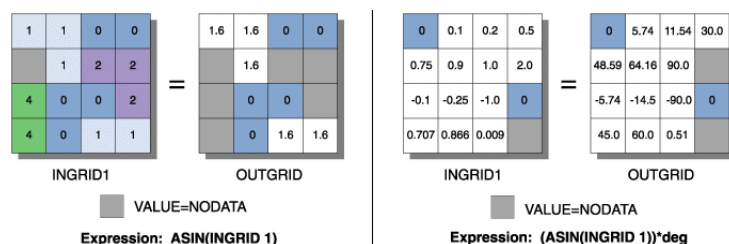
ACosH <in_raster_or_constant> <out_raster>



- 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.

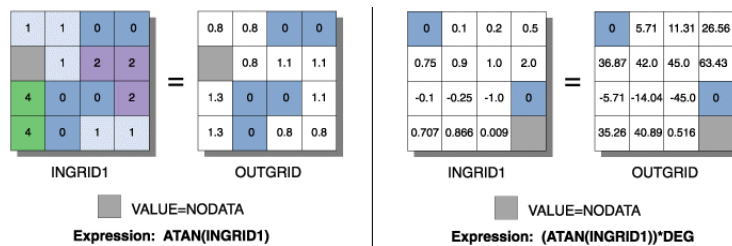
ASinH <in_raster_or_constant> <out_raster>



- 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.

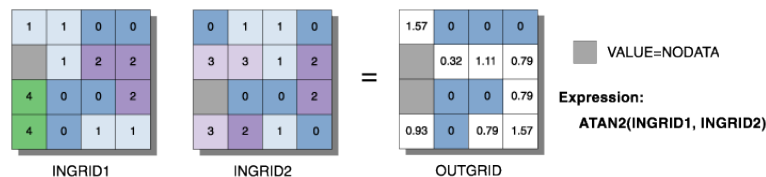
ATan <in_raster_or_constant> <out_raster>



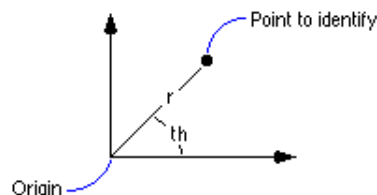
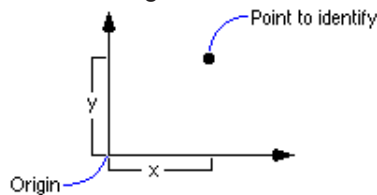
- The input values to the ATan function are interpreted as unitless.
- The output values from ATan are in radians. If degrees are desired, the resulting raster must be multiplied by the radians-to-degrees conversion factor of $180/\pi$, or approximately 57.296.

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>



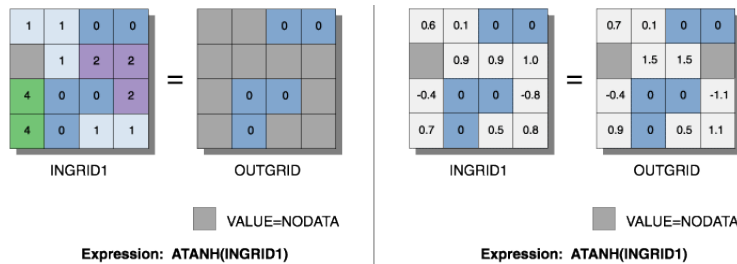
- The values of the first specified input are used as the numerator in the calculation of the tangent angle ($\tan@ = y/x$). The values of the second specified input are used as the denominator in the calculation of the angle.
- The output values from the ATan2 function are between $-\pi$ and π . The 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.

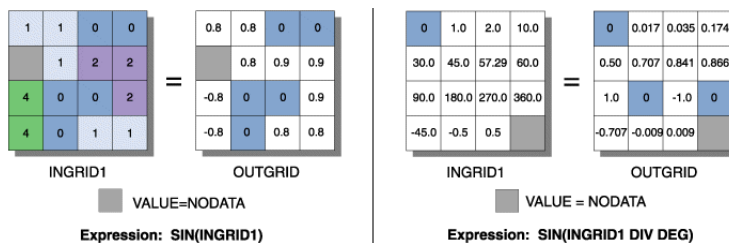
CosH <in_raster_or_constant> <out_raster>



- The input and output values in CosH are interpreted as unitless.

Sin: Calculates the sine of cells in an input raster dataset.

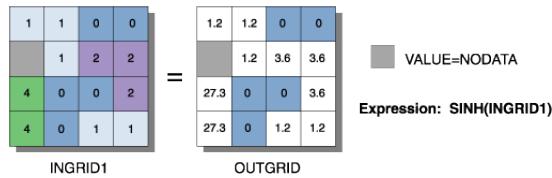
Sin <in_raster_or_constant> <out_raster>



- 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.

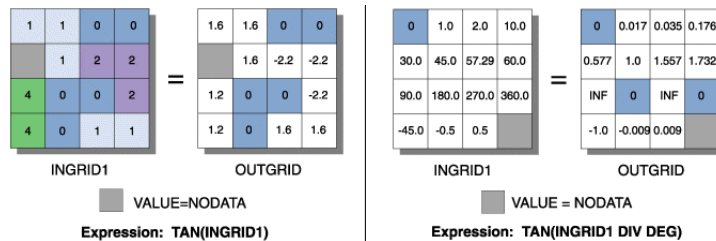
`SinH <in_raster_or_constant> <out_raster>`



- 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 is 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 zero 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 zero. If zero 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 that 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 three, class 4 with 11, and delete class 5.

Iso Cluster: Uses an isodata clustering algorithm to determine the characteristics of the natural groupings of cells in multidimensional attribute space and stores the results in an output ASCII signature file.

```
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 about 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 zero. If zero 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 similarly be assigned NoData.

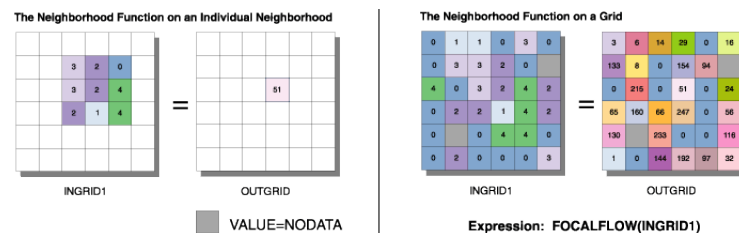
Filter: Performs a preset focal filter on a raster.

`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, 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 zero. 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 zero. For the other statistics, NoData is assigned.

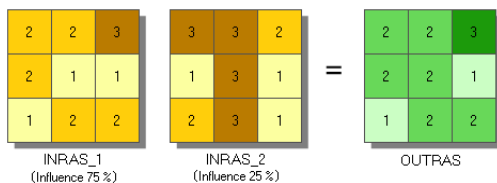


Overlay toolset

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.

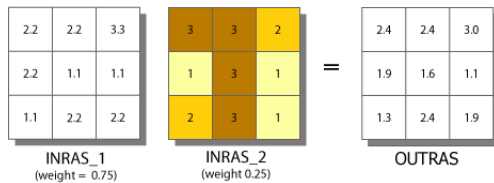
`weightedOverlay <raster{influence} {field} {remap};raster{influence} {field} {remap}...>`
`<out_raster>`



- 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 its percent influence. The weight is a relative percentage, and the sum of the percent influence weights must equal 100 percent.

Weighted Sum: Overlays several rasters multiplying each by their given weight and summing them together.

`weightedSum <Raster {Field} {weight};Raster {Field} {weight}...> <out_raster>`



- In the Illustration above, the cell values are multiplied by their weight factor, then added to create the output raster. For example, consider the top left cell. The values for the two inputs become $(2.2 * .75) = 1.65$ and $(3 * .25) = .75$. The sum of 1.65 and .75 is 2.4.
- Input rasters can be integer or floating point.
- The weight values can be any positive or negative decimal value. It is not restricted to be a relative percentage, or equal to 1.0.
- The weight will be applied to the specified field for the input raster. Fields can be, short or long integer, double, or float.
- The output raster will always be floating-point type.



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.

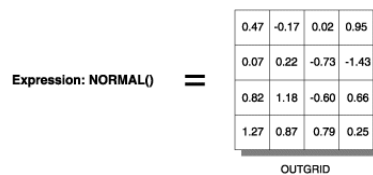
Create Constant Raster: Creates a raster dataset from a constant value.

`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.

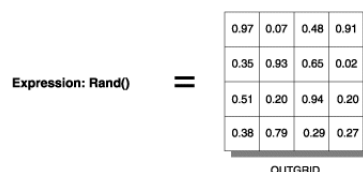
`CreateNormalRaster <out_raster> {cell_size} {extent}`



- 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 zero 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 where 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 zero 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 of a raster by zones of equal interval, equal area, or by natural breaks.

`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.



Solar Radiation toolset

Contains tools to perform solar radiation analysis.

Area Solar Radiation: Derives incoming solar radiation from a raster surface.

`AreaSolarRadiation <in_surface_raster> <out_global_radiation_raster> {latitude} {sky_size} {time_configuration} {day_interval} {hour_interval} {NOINTERVAL | INTERVAL} {z_factor} {FROM_DEM | FLAT_SURFACE} {calculation_directions} {zenith_divisions} {azimuth_divisions} {UNIFORM_SKY | STANDARD_OVERCAST_SKY} {diffuse_proportion} {transmittivity} {out_direct_radiation_raster} {out_diffuse_radiation_raster} {out_direct_duration_raster}`

- Because insolation calculations can be time consuming, it is important to be sure all parameters are correct. Calculation for a large digital elevation model can take hours, and a very large DEM could take days.
- The output radiation rasters will always be floating-point type and have units of watt hours per square meter (WH/m²). The direct duration raster output will be integer with unit hours.

Points Solar Radiation: Derives incoming solar radiation for specific locations in a point feature class or location table.

`PointsSolarRadiation <in_surface_raster> <in_points_feature_or_table> <out_global_radiation_features> {height_offset} {latitude} {sky_size} {time_configuration} {day_interval} {hour_interval} {NOINTERVAL | INTERVAL} {z_factor} {FROM_DEM | FLAT_SURFACE | FROM_POINTS_TABLE} {calculation_directions} {zenith_divisions} {azimuth_divisions} {UNIFORM_SKY | STANDARD_OVERCAST_SKY} {diffuse_proportion} {transmittivity} {out_direct_radiation_features} {out_diffuse_radiation_features} {out_direct_duration_features}`

- The input locations can be a feature class or table. The table can be an INFO table, a .dbf file, an Access table, or a text table file.
- When inputting locations by table, a list of locations must be specified with an x,y coordinate.
- For multiple-day time configurations, the maximum range of days is a total of one year (365 days, or 366 days for leap years).
- For within-day time configurations, the maximum range of time is one day (24 hours).

Solar Radiation Graphics: Derives raster representations of a hemispherical viewshed, sunmap, and skymap, which are used in the calculation of direct, diffuse, and global solar radiation.

`SolarRadiationGraphics <in_surface_raster> <out_viewshed_raster> {in_points_feature_or_table} {sky_size} {height_offset} {calculation_directions} {latitude} {time_configuration} {day_interval} {hour_interval} {out_sunmap_raster} {zenith_divisions} {azimuth_divisions} {out_skymap_raster}`

- Outputs are raster representations and are not maps that correspond to the outputs from the area or point solar radiation analysis. Rather, they are representations of directions in a hemisphere of directions looking upward from a given location. In a hemispherical projection, the center is the zenith, the edge of the circular “map” is the horizon, and the angle relative to the zenith is proportional to the radius. Hemispherical projections do not have a geographic coordinate system and have a bottom left corner of (0,0).
- It would not be practical to store viewsheds for all locations in a DEM, so when input locations are not specified, a single viewshed is created for the center of the input surface raster.
- The input locations table can be an INFO table, a .dbf file, an Access table, or a text file.
- Output graphic display rasters do not honor extent or cell size environment settings.
- One or two sun map rasters may be generated, depending on whether the time configuration includes overlapping sun positions throughout the year.

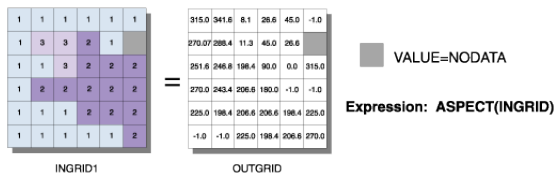


Surface toolset

Contains tools to analyze the surface of the shapes represented by the raster values.

Aspect: Identifies the downslope direction of the maximum rate of change in value from each cell to its neighbors.

`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 zero 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.

Contour With Barriers: Creates contours from a raster dataset surface using barrier features.

`ContourWithBarriers <in_raster> <out_contour_feature_class> {in_barrier_features} {POLYLINES | POLYGONS} {in_contour_values_file} {NO_EXPLICIT_VALUES_ONLY | EXPLICIT_VALUES_ONLY} {in_base_contour} {in_contour_interval} {in_indexed_contour_interval} {in_contour_list;in_contour_list...} {in_z_factor}`

- The inclusion of barrier features allows you to independently generate contours on either side of a barrier.

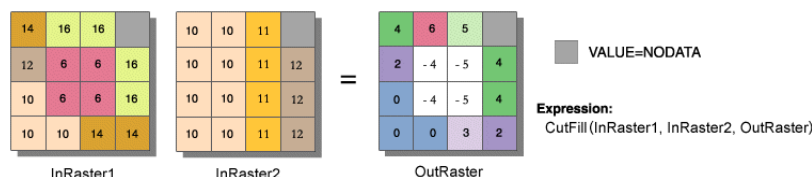
Curvature: Calculates the curvature of a raster surface, optionally including profile and plan curvature.

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 the volume and area of cut and fill locations.

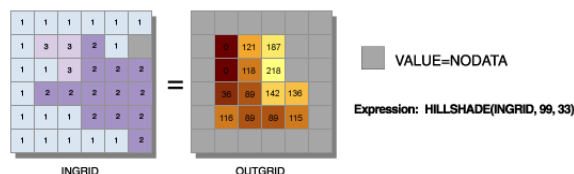
CutFill <in_before_surface> <in_after_surface> <out_raster> {z_factor}



- 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.

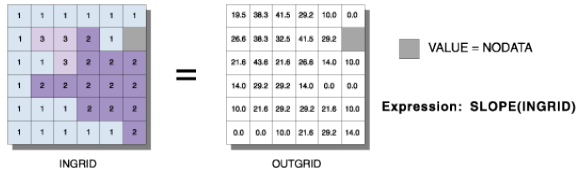
Observer Points: Identifies exactly which observer points are visible from each surface location.

`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 zero 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.
- The input zone raster must be integer.

Zonal Geometry: Calculates for each zone in a dataset the specified geometry measure (area, perimeter, thickness, or the characteristics of ellipse).

`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 for each zone in a dataset the geometry measures—for example, area, perimeter, thickness, and the characteristics of ellipse—and reports the results as a 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 settings. 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 zero 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 zero in the east (horizontal, to the right) and going through 90 when the major axis is vertical.
- An example of the Zonal Geometry As Table on the following input:



This would populate the output table as follows:

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	...



Tracking Analyst toolbox

This toolbox provides tools for preparing and loading temporal data into an ArcGIS application.

Concatenate Date and Time Fields: Concatenates two separate date and time fields into a new field that is added to the same feature dataset.

`ConcatenateDateAndTimeFields <feature_class> <date_field> <time_field> <output_field>`

- Input values must be text type.
- This tool adds a new text field to the input feature dataset.
- If you are working with a shapefile that contains a date field with the date only (for example, 03/09/2006), you must calculate the information into a new field that is of type text before you can concatenate it with a time field.

Make Tracking Layer: Adds a feature dataset as a new tracking layer to your ArcMap session.

`MakeTrackingLayer <feature_class> <date_field> {track_field} <output_name> {H:MM:SS TT | HH:MM:SS TT | H:MM:SS | HH:MM:SS} {M/D/YYYY | M/D/YY | MM/DD/YY | MM/DD/YYYY | YY/MM/DD | YYYY-MM-DD | DD-MMM-YY} {AM} {PM}`

- The date_field must contain both the date and time for the event.
- If the date field is string type, then you must specify the format of date and time.

Index

Numbers

3D Analyst toolbox	109
3D ASCII file	
converting to feature class.....	109
3D files	
importing	109
3D layer	
converting to feature class.....	109

A

Abs command	160
Absolute value. <i>See Abs command</i>	
ACos command.....	173
ACosH command.....	173
AddCADFields command.....	19
AddCodedValueToDomain command	49
AddColormap command.....	77
AddFeatureClassToTerrain command.....	123
AddFeatureClassToTopology command.....	81
AddField command.....	57
AddFieldToAnalysisLayer command	133
AddGlobalIDs command	63
AddIndex command.....	64
Adding. <i>See also Plus command</i>	
attribute index.....	64
CAD fields.....	19
coded value to domain	39
color maps.....	77
indexes.....	64
items	78
joins	65
rules to topology.....	81
spatial indexes	64
subtypes.....	79
x,y coordinates	12
AddLocations command	133
AddRepresentation command.....	85
Address	
geocoding	85
locator	
creating	86
deleting	86
rebuilding	86
AddressBuild. <i>See RebuildGeocodingIndex command</i>	
AddressCreate. <i>See CreateAddressLocator command</i>	
AddressMatch. <i>See GeocodeAddresses command</i>	
AddressParse. <i>See StandardizeAddresses command</i>	
AddRouteMeasure. <i>See LocateFeaturesAlongRoutes command</i>	
AddSubtype command.....	79
AddTerrainPoints command	124

AddTerrainPyramidLevel command.....	124
AddXY command	2
Adjust. <i>See Transform command</i>	
ADRGGrid. <i>See CopyRaster command</i>	
Advanced Tiger Conversion. <i>See TigerTool command</i>	
Aggregate	
command	147
Aggregate Polygons. <i>See AreaAggregate command</i>	
AggregatePolygons command	79
Analysis	
toolbox.....	76
toolset	133
Analyze command	80
Analyzing	
hot spots	34
Analyzing Patterns toolset	31
And	
bitwise	165
Boolean	167
combinatorial.....	168
Annotation	
importing from CAD.....	21
importing from coverage.....	65
Append command	35
ApplySymbologyFromLayer command	66
ArcDXF. <i>See ExportCAD command</i>	
ArcEditor	
licensing	2
ArcIGDS. <i>See ExportCAD command</i>	
ArcInfo	
licensing	47
ArcInfo workspace	
creating.....	84
ArcSection. <i>See CreateRoutes command</i>	
ArcShape. <i>See FeatureClassToShapefile command</i>	
ArcTiger. <i>See TigerArc command. See TigerTool command</i>	
ArcTIN. <i>See Editing: TINs</i>	
AreaAggregate. <i>See AggregatePolygons command</i>	
AreaSolarRadiation command	183
Arguments.....	57
ASCII	
converting to raster.....	109
Ascii3DToFeatureClass command.....	49
ASCIIGrid. <i>See ASCII: converting to raster</i>	
ASin command.....	173
ASinH command.....	174
Aspect command.....	121, 184
ATan2 command	174
ATan command	174
ATanH command	175
Attributes	
joining tables	85

B

Band Collection Statistics. <i>See</i> <i>BandCollectionStats</i> command	
BandCollectionStats command	176
Base 2. <i>See</i> <i>Exp2</i> command. <i>See also</i> <i>Log2</i> command	
Base 10. <i>See</i> <i>Exp10</i> command. <i>See</i> <i>Log10</i> command	
Base e. <i>See</i> <i>Exp</i> command	
Basic Tiger Conversion. <i>See</i> <i>TigerArc:</i> command	
Basin command	151
Batch	
projecting	69
BatchBuildPyramids command	77
BatchCalculateStatistics command	77
BatchProject command	69
Bitwise	
And 165	
Complement. <i>See</i> <i>BitwiseNot</i> command	
Exclusive Or	167. <i>See also</i> <i>BitwiseXOr</i> command
left shift	165
Not 166	
Or 166	
right shift	166
toolset	165
BitwiseAnd command	165
BitwiseLeftShift command	165
BitwiseNot command	166
BitwiseOr command	166
BitwiseRightShift command	166
BitwiseXOr command	167
BlockMajority. <i>See</i> <i>BlockStatistics</i> command	
BlockMax. <i>See</i> <i>BlockStatistics</i> command	
BlockMean. <i>See</i> <i>BlockStatistics</i> command	
BlockMedian. <i>See</i> <i>BlockStatistics</i> command	
BlockMin. <i>See</i> <i>BlockStatistics</i> command	
BlockMinority. <i>See</i> <i>BlockStatistics</i> command	
BlockRange. <i>See</i> <i>BlockStatistics</i> command	
BlockStatistics command	179
BlockStd. <i>See</i> <i>BlockStatistics</i> command	
BlockSum. <i>See</i> <i>BlockStatistics</i> command	
BlockVariety. <i>See</i> <i>BlockStatistics</i> command.	
<i>See</i> <i>BlockStatistics</i> command	
Boolean	
And	167
Complement. <i>See</i> <i>BooleanNot</i> command	
Exclusive Or	168
Not	167
or	168
BooleanAnd command	167
BooleanNot command	167
BooleanOr command	168
BooleanXOr command	168
BoundaryClean command	147
Buffer	
command	8, 42
using multiple rings	9

Build command	38
Building	
pyramids	77
raster attribute tables	77
BuildNetwork command	135
BuildPyramids command	77
BuildRasterAttributeTable command	77
BuildSta. <i>See</i> <i>CellStatistics</i> command	
BuildTerrain command	124

C

Cache	
clearing	46
creating for Globe Server	94
creating for Map Server	93, 94
deleting for Globe Server	93
deleting for Map Server	93
tiling scheme	94
Caching	
toolset	93
CAD	
importing	51
importing annotation	21
CalculateAreas command	104
CalculateDistanceBand command	104
CalculateEndDate command	70
CalculateField command	59
CalculateGeodesicAngle command	11
CalculateLineCaps command	51
CalculateLocationFields command	133
CalculateLocations command	133
CalculatePolygonMainAngle command	63
CalculateRepresentationRule command	70
CalculateStatistics command	77
CalculateValue command	87
Calculating	
kernel density	140
spatial grid index	64
CAnd. <i>See</i> <i>CombinatorialAnd</i> command	
Cell groupings	
using isodata clustering	178
Cells	
get value	78
CellStatistics command	157
Centerline	
creating	47
CentroidLabels. <i>See</i> <i>FeatureToPoint</i> command	
ChangePrivileges command	80
ChangeTerrainReferenceScale command	124
CheckGeometry command	47
Checking in	
from Delta geodatabases	36
to ArcSDE	46
ClassProb. <i>See</i> <i>ClassProbability</i> command	

ClassProbability command	176	ConvertRemap. <i>See ReclassByASCIIFile command</i>	
ClassSig. <i>See CreateSignatures command</i>		ConvertSpatialWeightsMatrixToTable command	105
Cleaning		Coordinates	
boundaries	147	adding x,y	76
ClearWorkspaceCache command		Copy command	28
Clip command	51	CopyFeatures command	32
Cluster and outlier analysis	103	Copying	
Clustering		feature classes	33
high/low	35	feature datasets	42
ClustersOutliers command	70	features	33
ClustersOutliersRendered command	103	raster catalog items	72
Cluster tolerance		rows in a table	80
setting	82	tables	42
CollapseDualLinesToCenterline command	17	CopyRasterCatalogItems command	72
CollapseDualLineToCenterline command		CopyRaster command	73
CollectEventsRendered command	103	CopyRows command	80
Color maps		COr. <i>See CombinatorialOr command</i>	
adding	77	Corridor command	175
deleting	78	Cos command	42
Combinatorial		CosH command	175
and	168	Cost	
exclusive or	169	calculating least accumulative distance	142
or	169	least accumulative	142
CombinatorialAnd command	168	least accumulative distance	
CombinatorialOr command	169	path	142
CombinatorialXOr command	169	least accumulative path	
Combine command	157	define neighbor cell	141
Combining features. <i>See Append command</i>		least accumulative source	144
Compact command	45	CostAllocation command	141
CompareReplicaSchema command	45	CostBackLink command	141
Comparing		CostDistance command	142
features	45	CostPath command	142
files	46	Counting	
rasters		table rows. <i>See GetCount command</i>	
tables	41	CountRenderer command	103
TINs	46	Coverages	
CompositeBands command	75	creating	85
Composite Features toolset	58	erasing	19
ConcatenateDateAndTimeFields command	189	exporting as text file	80
Con command	139	generating topology	66
Conditional toolset	139	importing from DLG	50
Conflicts		importing from S57	36
finding	126	importing from SDTS	51
Contour command		importing from VPF	80
Contours		projecting	51
creating	184	setting tolerances	80
with barriers	59	toolbox	43
ContourWithBarriers command	53	updating	89
Conversion		updating attribute table and topology	
toolbox	137	CreateAddressLocator command	
toolset	59, 34	CreateArcInfoWorkspace command	84
ConvertDiagram command	34	CreateCADXData command	137
ConvertImage. <i>See CopyRaster command</i>		Create command	181
Converting		CreateConstantRaster command	52
to feature class	34	CreateCustomGeoTransformation command	68
to shapefile	76	CreateDiagram command	93

CreateDomain command		DecimateTinNodes command	126
CreateFeatureClass command	75	DefineProjection command	85
CreateFeatureDataset command	84	DeleteAddressLocator command	50
CreateFishnet command	48	DeleteCodedValueFromDomain command	73
CreateFolder command	84	DeleteColormap command	78
CreateLabels command	11	Delete command	60
CreateMapServerCache command	67	DeleteDomain command	53
CreateMobileBasemap command	181	DeleteFeatures command	54
CreateNormalRaster command	76	DeleteField command	58
CreateOrthoCorrectedRasterDataset command	76	DeleteGlobeServerCache command	93
CreatePersonalGDB command	84	DeleteMapServerCache command	73
CreateRelationshipClass command	78	DeleteRasterAttributeTable command	78
CreateReplica command	137	DeleteRasterCatalogItems command	73
CreateReplicaFootPrints command	123	DeleteRows command	80
CreateReplicaFromServer command	80	DeleteVersion command	83
CreateRoutes command	124	Deleting	
CreateSchematicFolder command	177	coded values from domains	59
CreateSignatures command	66	domains	59
CreateSpatialReference command	69	feature classes	60
CreateTable command	80	feature datasets	60
CreateTerrain command	124	features	54
CreateThiessenPolygons command	8	fields in a table	58
CreateTIN command	125	geodatabase versions	83
CreateTopology command	82	items in a raster catalog	72
CreateTurnFeatureClass command		rows in a table	80
CreateUnderpass command		tables	60
CreateVersion command	83	DelineateTinDataArea command	125
CreateWorkspace. <i>See CreateArcInfoWorkspace command</i>		DEM	
Creating		converting to raster	155
ArcInfo workspaces	84	DEMGrid. <i>See DEM: converting to raster</i>	
CAD x data	79	DEMLattice. <i>See DEM: converting to raster</i>	
multiband raster datasets	75	Dendrogram command	177
personal geodatabases	84	Density	
spatial references	69	calculating kernel	140
table to relationship classes	79	calculating using line features	140
CuldeSacMasks command	184	calculating using point features	140
CXOr. <i>See CombinatorialXOr command</i>		toolset	140
D		DetectGraphicConflict command	37
DarcyFlow command	149	DFADArc. <i>See Feature class: converting to coverage</i>	
DarcyVelocity command	150	Diagrams	
Data		converting	47
compressing	58	creating	137
selecting	61	updating	137
Database toolset	46	Updating	137
Data Comparison toolset	45	Diff command	169
Data Interoperability toolbox	129	DirectionalDistribution command	99
Data Management toolbox	34	DirectionalMean command	133
Data Management toolset	45	Directions command	87
Datasets		Disconnected Editing toolset	87
deleting	60	DisperseMarkers command	33
renaming	58	Dissolve command	48
uncompressing	40	DissolveRouteEvents command	118
DBaseInfo. <i>See TableToGeodatabase command</i>		Dissolving	
DBMSInfo. <i>See TableToGeodatabase command</i>		route events	50
DeautomateGeocodingIndexes command	85	Distance toolset	141
		Distributed Geodatabase toolset	50

Divide command.....	50, 160. <i>See also Mod command</i>
DLG	
importing to coverage	39
DLGArc command.....	49
Domain	
creating.....	39
creating from table	51
deleting.....	39
removing from field.....	12
Drainage. <i>See Hydrology toolset</i>	
DropIndex command. <i>See RemoveIndex command</i>	
DTEDGrid. <i>See CopyRaster command</i>	
DXFArc. <i>See ImportCAD command</i>	
DXFINFO. <i>See ImportCAD command</i>	
E	
Editing	
TINs	126
EditSig. <i>See EditSignatures command</i>	
EditSignatures command	177
EditTin command.....	126
Eliminate command	37, 62
Ellipse	2
Envelopes	
creating polygons from	54
EqualityTest. <i>See EqualTo command</i>	
EqualTo command	170
EqualToFrequency command	158
Erase command.....	6, 28
ESRI Metadata Translator.....	18
ESRITranslator command.....	18
ETAKArc. <i>See FeatureClassToCoverage command</i>	
EucAllocation command.....	143
EucDirection command	143
EucDistance command.....	143
Euclidean	
allocating.....	143. <i>See also EucAllocation command</i>
direction.....	143. <i>See also EucDirection command</i>
distance.....	143. <i>See also EucDistance command</i>
EventArc. <i>See MakeRouteEventLayer command</i>	
EventPoint. <i>See MakeRouteEventLayer command</i>	
Events	
dissolving for routes.....	88
EventTransform. <i>See TransformRouteEvents command</i>	
Exclusive Or	
bitwise	167
Boolean	169
combinatorial.....	161
Exp2 command	161
Exp10 command	147
Expand command.....	160
Exp command	161
Exponential. <i>See Exp command</i>	
base 2.....	161
base 10.....	160
base e.....	161

ExportAcknowledgementMessage command.....	48
ExportCAD command	31
Export command.....	48
Exporting	
feature attribute to ASCII	105
to CAD	47
to Delta geodatabase	48
to interchange file.....	32
to VPF	72
ExportRasterCatalogPaths command.....	73
ExportRasterWorldFile command.....	78
ExportToDelta command	48
ExportXYv command.....	105
ExtractByAttributes command.....	145
ExtractByCircle command.....	145
ExtractByMask command.....	146
ExtractByPoints command.....	145
ExtractByPolygon command	146
ExtractByRectangle command.....	145
Extraction toolset	75
Extract toolset	146
ExtractValuesToPoints command	125
F	
Feature class	
adding to terrain	123
combining.....	59
comparing.....	20
converting to coverage	20
converting to feature class.....	20
converting to shapefile	20
creating.....	6
intersecting	78
remove from topology.....	82
toolset	7
updating attributes and geometry	19
FeatureClassToCoverage command.....	20
FeatureClassToFeatureClass command	20
FeatureclassToShapefile command	25
FeatureOutlineMasks command	101
Features	
buffering.....	22
center of concentration.....	100
centrally located	99
concentration or dispersion	100
converting to raster.....	23
copying.....	6
deleting.....	54
dissolving	62
eliminating.....	62
erasing	52
extracting.....	7
locating along routes	54
toolset	53
unioning.....	67
FeatureToLine command	54

- FeatureToNetCDF command 22
 - Feature toolset 69
 - FeatureToPoint command 56
 - FeatureToPolygon command 55
 - FeatureToRaster command 23
 - FeatureVerticesToPoints command 57
 - Fields
 - adding 57
 - assigning default to 57
 - calculating 37
 - joining 65
 - FileCompare command 67
 - Fill command 152
 - Filter command 179
 - FindConflicts command 23
 - Float
 - command 119, 161
 - convert to raster 181
 - FloatGrid. *See FloatToRaster command*
 - Flow. *See DarcyFlow command*
 - FlowAccumulation command 152
 - FlowDirection command 152
 - FlowLength command 153
 - FMod. *See Mod command*
 - FocalFlow command 179
 - 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 179
 - FocalStd. *See FocalStatistics command*
 - FocalSum. *See FocalStatistics command*
 - FocalVariety. *See FocalStatistics command*
 - From Feature Class toolset 109
 - From File toolset 109
 - From Raster toolset 110
 - From Terrain toolset 111
 - From TIN toolset 111
 - Functional Surface toolset 114
- G**
- GACalculateZValue command 131
 - GACreateGeostatisticalLayer command 131
 - GAGetModelParameter command 132
 - GALayerToContour command 131
 - GALayerToGrid command 131
 - GALayerToPoints command 131
 - GAMovingWindowKriging command 132
 - GANeighborhoodSelection command 132
 - GASemivariogramSensitivity command 132
 - GASetModelParameter command 132
 - GaussianGeostatisticalSimulations command 131
 - Generalization toolset 58, 147
 - Generalize. *See Line: simplification. See Lines: simplifying*
 - General toolset 59
 - Generate command 8
 - GenerateGlobeServerCache command 94
 - GenerateMapServerCache command 89
 - GenerateMapServerCacheTilingScheme command 85
 - GenerateMobileServiceCache command 102
 - GenerateNearTable command 102
 - GenerateNetworkSpatialWeights command 101
 - GenerateSpatialWeightsMatrix command 101
 - GeocodeAddresses command 86
 - Geocoding
 - addresses 81
 - deautomating indexes 53
 - index
 - rebuilding 86
 - toolbox 11
 - Geodatabase
 - creating 84
 - Geostatistical Analyst toolbox 131
 - Geostatistical layer
 - converting to contours 131
 - converting to grid 131
 - converting to points 131
 - creating 131
 - GreaterThan command 170
 - GreaterThanEqual command 170
 - GreaterThanFrequency command 158
 - 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*
 - 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 149
 - dispersion of point 151
 - toolset 149
- H**
- HDF files
 - working with 76
 - HighestPosition command 158
 - HighLowClustering command 97

HillShade command..... 122, 185
 HotSpot command 99
 HotSpotsRendered command..... 103
 Hydrodynamic dispersion. *See PorousPuff command*
 Hydrology toolset..... 151

I

IDEdit command..... 41
 Identity command 6, 28
 IDs
 updating..... 136
 IDW command..... 115, 154
 IGDSArc. *See ImportCAD command*
 IGDSInfo. *See ImportCAD command*
 ImageGrid. *See CopyRaster command*
 Import3DFiles command 109
 ImportCADAnnotation command 21
 ImportCAD command 33
 Import command..... 21
 Importing
 CAD annotation 21
 coverage annotation 21
 from CAD..... 33
 from S57..... 34
 from SDTS 34
 from VPF 49
 ImportMessage command..... 49
 IncreaseMaximumEdges command 135
 Index
 adding 64
 adding spatial index..... 64
 automating geocoding index 39
 dropping 39
 item 86
 rebuilding for geocoding 39
 removing 65
 Indexes toolset 39
 IndexItem command 39. *See also AddIndex command*
 InfoDBASE. *See TableToDBASE command*
 InList command 171
 Int command 119, 161
 Integer
 converting from. *See Int command*
 Integrate command..... 129
 InterpolatePolyToPatch command 113
 Interpolating
 a raster dataset..... 154
 calculating the trend on a point dataset..... 157
 raster surfaces..... 113, 154
 z-values 114
 Interpolation toolset 154
 Intersecting
 coverages..... 115
 IntersectingLayersMasks command..... 39
 Inverse distance weighted 39, 154
 IsNull command..... 171

IsoCluster command 178

J

JoinField command..... 65
 Joining
 attribute tables 65
 fields 65
 info tables 39
 JoinItem command..... 39
 Joins
 adding 65
 removing 66
 toolset 39, 65

K

KernelDensity command 140
 Kill. *See Delete command*
 KML
 create from layer 22, 113
 create from map..... 22, 114
 Kriging. *See Krige command. See Kriging command*
 Kriging command 116, 155

L

LAS
 converting to multipoint..... 110
 LASToMultipoint command..... 110
 LatticeClip. *See ExtractByPolygon command*
 LatticeContour. *See Contour command*
 Layer
 make from feature 66
 make from query table 67
 Layer3DToFeatureClass command..... 109
 Layers and Table Views toolset..... 66
 LayerToKML command..... 2, 113
 LessThan command 171
 LessThanEqual command..... 2
 LessThanFrequency command 171
 Licensing
 tools A-1
 Linear Referencing toolbox 140
 LineDensity command..... 35
 LineGrid. *See FeatureToRaster command*
 LineOfSight command..... 35
 Lines
 converting
 coverage to region 55
 coverage to route 38
 splitting at vertices 179
 LineStatistics command..... 180
 LineStats. *See LineStatistics command*
 Ln command 157
 Local toolset..... 87
 Locating
 features along routes 162

- Log2 command 162
- Logarithm. *See also Ln command*
 - base 2 162
 - base 10 162
 - base e 162
 - natural 167
- Logical toolset 119
- Lookup command 120
- M**
- Majority. *See CellStatistics command*
- MajorityFilter command 147
- MakeClosestFacilityLayer command 133
- MakeFeatureLayer command 66
- MakeImageServerLayer command 67
- MakeNetCDFFeatureLayer command 91
- MakeNetCDFRasterLayer command 91
- MakeNetCDFTableView command 91
- MakeODCostMatrixLayer command 134
- MakeQueryTable command 67
- MakeRasterCatalogLayer command 67
- MakeRasterLayer command 67
- MakeRouteEventLayer command 88
- MakeRouteLayer command 134
- MakeServiceAreaLayer command 134
- MakeTableView command 67
- MakeTrackingLayer command 189
- MakeVehicleRoutingProblemLayer command 134
- MakeWCSLayer command 67
- MakeXYEventLayer command 68
- ManageGlobeServerCacheTiles command 94
- ManageMapServerCacheScales command 94
- ManageMapServerCacheTiles command 95
- Map Algebra
 - single output 160
 - statements 160
 - toolset 160
- Mapping Clusters toolset 11
- MapToKML command 11
- Mask
 - creating cul-de-sac 11
 - creating from feature outline 12
- Masking toolset 18
- Masks
 - outlining features 18
- MasksFromFeatureOutlineMasks command 59
- MasksFromIntersectingLayersMasks command 160
- Math toolset 59
- Max. *See CellStatistics command*
- Maximum Likelihood Classification. *See MLClassify command*
- MDPublisher command 101
- Mean. *See CellStatistics command*
- MeanCenter command 100
- MeasureRoute. *See CreateRoutes command*
- Measuring Geographic Distributions toolset 99
- Med. *See CellStatistics command*
- MergeBranch command 60
- MigrateStorage command 89
- Min. *See CellStatistics command*
- Minimum curvature. *See Spline command*
- Minority. *See CellStatistics command*
- Mirror command 70
- Mobile toolbox 162
- Mod command 163
- Model tool 55
- MultiDistanceSpatialClustering command 159
- Multipart feature
 - converting to single-part feature 109
- Multiply. *See Times command*
- N**
- NaturalNeighbor command 116, 155
- Near command 9, 30
- Nearest neighbor 97
- Negate command 163
- Neighborhood selection 132
- Neighborhood toolset 179
- NetCDF
 - layers 91
- Network Analyst toolbox 133
- Network Dataset toolset 135
- Networks
 - building 135
- Nibble command 148
- Normal. *See CreateNormalRaster command*
- Not
 - bitwise 166
 - Boolean 167
- NotEqual command 172
- O**
- ObserverPoints command 123, 186
- Or
 - bitwise 166
 - Boolean 168
 - combinatorial 169
- OrdinaryLeastSquares command 102
- Outlier and cluster analysis 98, 103
- Over command 172
- Overlaying
 - route events 88
- OverlayRouteEvents command 6
- Overlay toolset 14
- P**
- Pan-sharpening
 - raster data 76
- Parameters 2
- optional 2

Particle	
calculating path	150
ParticleTrack command	150
PathAllocation command	144
PathBackLink command	144
PathDistance command	144
Pick command	139
PivotTable command	81
Pixels	
get value	78
Plus command	119, 163
Point	
calculating at surface. <i>See SurfaceSpot command</i>	
PointDensity command	140
PointDistance command	10, 30
PointFileInformation command	9
PointGrid. <i>See FeatureToRaster command</i>	
Points	
calculating at surface. <i>See SurfaceSpot command</i>	
calculating distance between	10
creating from features	36
creating from vertices	35
PointsSolarRadiation command	183
PointStatistics command	180
PointToRaster command	24
Polygons	
aggregating	8
converting coverages to regions	23
converting to lines	56
merging	55
PolyGrid. <i>See FeatureToRaster command</i>	
Popularity command	159
PopulateAlternateIDFields command	135
PorousPuff command	151
PostVersion command	83
Power command	163
PrincipalComponents command	179
PrinComp. <i>See PrincipalComponents command</i>	
Privileges	
changing	80
Project command	8
ProjectDefine. <i>See DefineProjection command</i>	
ProjectGrid. <i>See ProjectRaster command</i>	
Projecting	69
Projections and Transformations toolset	68
ProjectRaster command	70
Pyramids	
adding to terrain	124
building for rasters	77

Q

QuickExport command	129
QuickImport command	129

R

Range. <i>See CellStatistics command</i>	
Rank command	159
Raster attribute tables	
building	77
Raster catalogs	
creating	72
deleting items	74
export paths	73
fix paths	73
from raster datasets	74
making layers	67
Raster Catalog toolset	73
RasterCatalogToRasterDataset command	74
Raster Creation toolset	181
Raster datasets	
calculating accumulated flow	152
calculating band statistics	77
calculating drainage basins	151
calculating flow direction	152
calculating flow distance	153
calculating slope	123, 186
calculating statistics	77, 179
calculating surface length	17
calculating surface values	114
calculating surface visibility	17
calculating surface volume	115
calculating weighted overlay	180
changing cell values. <i>See Reclassify command</i>	
changing scale	70
clipping	75
combining	157
comparing	70
converting from TINs	113
converting to ASCII	
converting to linear network features	154
converting to points	68
converting to polygons	45
converting to polylines	112
converting values to float	161
converting values to integer. <i>See Int command</i>	
creating	115, 118, 181
creating ASCII file of selected cells	146
creating contours	121, 184
creating surface. <i>See Raster Interpolation toolset</i>	
creating TINs	125
creating using elevation data	156, 155
extracting cells	
using a mask	145
using attributes	145
using points	145
within a circle	145
within a polygon	146
within a rectangle	146

Raster datasets (continued)	
extracting cell values	
using points	145
interpolating	115, 126, 154
majority filter	147
making layers	67
mosaicking	74
performing a classification	178, 179
principal components	179
projecting	70
reclassifying	120, 121, 182
recording least cost path	142
reducing resolution	147
replacing cell values. <i>See Nibble command</i>	
resampling	76
rotating	71
solar radiation	183
surface	
calculating viewable areas	123, 186
removing sinks	152
surface flow values	179
to raster catalogs	74
transforming	70
viewshed	123, 186
Raster Dataset toolset	73
RasterDomain command	73
Raster Interpolation toolset	17
Raster Math toolset	122
Raster Processing toolset	75
Raster Properties toolset	77
Raster Reclass toolset	120
Rasters	
classification	72
Raster Surface toolset	121
RasterTIN command	111
RasterToASCII command	120
RasterToFloat command	178
RasterToGeodatabase command	72
RasterToNetCDF command	110
Raster toolset	70
RasterToOtherFormat command	25
RasterToPoint command	67
RasterToPolygon command	35
RasterToPolyline command	35
RebuildAddressLocator	86
RebuildGeocodingIndex command	86
Rebuilding	
geocoding indexes	119
Reclass. <i>See ReclassByTable command.</i>	
toolset	182
ReclassByASCIIFile command	120, 182
ReclassByTable command	79, 182
Reclassify command	121, 182
ReconcileVersion command	83
Rectify. <i>See ProjectRaster command</i>	
Reducing database sizes	123
RegionGroup command	148
Regions	
converting to polygon coverages	50
grouping	148
RegisterAsVersioned command	83
Relationship Classes toolset	78
RematchAddresses	124
RemoveDomainFromField command	64
RemoveFeatureClassFromTerrain command	125
RemoveFeatureClassFromTopology command	82
RemoveIndex command	65
RemoveJoin command	66
RemoveOverride command	64
RemoveRuleFromTopology command	82
RemoveSpatialIndex command	65
RemoveSubtype command	79
RemoveTerrainPoints command	125
RemoveTerrainPyramidLevel command	125
Removing	
attribute indexes	65
feature classes from topologies	82
joins	66
rules from topologies	82
spatial indexes	65
Rename command	14
Rendering	
collecting events and	103
counts	103
with hot spot analysis	103
Renode command	72
Renumbering	
nodes	27
Representation markers	
disperse	68
Representations	
creating overpasses	68
update overrides	87
Rescale command	70
Reselect command	88. <i>See also Select command</i>
Rotate command	71
RoundDown command	164
RoundUp command	164
Route layer	
create	77
Routes	
calibrating	78
creating	78
S	
S57Arc command	33
Sample command	146
SaveToLayerFile command	68
Saving	
layer files	68
Scale	
changing	70

Schematics toolbox	137	SpatialJoin command	116
Script tool	32	Spatial reference	
SDTSExport command	34	creating	69
SelectBox. <i>See ExtractByRectangle command</i>		upgrading	47
SelectByDimension command	5	Spatial Statistics toolbox	101
SelectCircle. <i>See ExtractByCircle command</i>		Spline command	116, 155
Select command	13	SplineWithBarriers command	9
SelectFeatureByOverride command	66	Split command	9
Selecting		SplitLine command	9
by attributes	68	SQR. <i>See Square command</i>	
by locations	68	SQRT. <i>See SquareRoot command</i>	
data	61	Square command	70
layers by attributes	68	SquareRoot command	165
layers by locations	68	Standard deviational ellipse	99
SelectMask. <i>See ExtractByMask command</i>		Statistics	
SelectPoint. <i>See ExtractByPoints command</i>		calculating for a block	76
SelectPolygon. <i>See ExtractByPolygon command</i>		calculating summary	10
SelectPolygon command. <i>See ExtractByPolygon command</i>		frequency	10
Semivariogram sensitivity	78	of a cell	75
Server toolbox	76	StreamLink command	153
Service area layer		StreamShape. <i>See StreamToFeature command</i>	
create	14	StreamToFeature command	7
SetCADAlias command	76	Subdatasets	
SetClusterTolerance command	82	extracting	76
SetDefaultSubtype command	79	Subtract. <i>See Minus command</i>	
SetNull command	50	Subtypes toolset	79
SetRepresentationControlPointByAngle command	15	Sum. <i>See CellStatistics command</i>	
Shaded relief. <i>See Hillshade command</i>		Summary Statistics. <i>See Statistics: command</i>	
ShapeGrid. <i>See FeatureToRaster command</i>		Surface	
Shift command	71	aspect	185
Shrink command	148	slope	123
Signatures files		T	
creating	176	TableCompare command	45
creating probability layers	177	Tables	
editing	38	calculating frequency	10
SimplifyBuilding command	62	comparing	45
SimplifyLineOrPolygon command	175	converting or copying to tables	22
Sin command	120	copying	59
SingleOutputMapAlgebra command	175	creating	80
SinH command	176	creating from domains	50
Sink command	122	creating pivot tables	81
Slice command	121, 183	deleting	60
SnapPour. <i>See SnapPourPoint command</i>		importing to geodatabases	22
SnapPourPoint command	98	making table views	67
SnapPout. <i>See SnapPourPoint command</i>		renaming	60
Solar radiation		selecting	5
deriving incoming	183	TableSelect command	41
representing	183	Tables toolset	20
toolset	134	TableToDBASE command	21
SolarRadiationGraphics command	51	TableToDomain command	51
Solve command	135	TableToNetCDF command	92
Spatial Analyst toolbox	67	Table toolset	80
SpatialAutocorrelation command	46	TableToRelationshipClass command	79
Spatial grid index		TableToTable command	22
calculating	115	TabulateArea command	186

Tan command.....	176	To KML toolset.....	22, 113
TanH command.....	176	Tolerance	
Temporary layers		command.....	11
making. <i>See Layers and Table Views toolset</i>		setting cluster tolerances.....	82
Terrains		Tolerances toolset.....	1
adding feature class.....	123	Tool	
adding pyramid.....	124	licensing.....	A-1
building.....	124	Toolbox	
creating.....	124	3D Analyst.....	109
points.....	124, 125	Analysis.....	87
reference scale.....	124	Cartography.....	1
remove pyramid.....	125	Conversion.....	89
Terrain toolset.....	123	Coverage.....	91
TerrainToRaster command.....	111	Data Interoperability.....	129
TerrainToTin command.....	111	Data Management.....	1
Test command.....	172	described.....	93
Thiessen		Geocoding.....	1
polygons.....	32	Geostatistical Analyst.....	131
Thiessen command.....	33	Linear Referencing.....	97
Thin command.....	149	Mobile.....	34
Times command.....	120, 165	Multidimension.....	27
TINArc. <i>See FeatureClassToCoverage command</i>		Network Analyst.....	133
TinAspect command.....	127	Schematics.....	137
TinContour command.....	127	Server.....	35
TIN Creation toolset.....	125	Spatial Analyst.....	139
TinDifference command.....	127	Spatial Statistics.....	31
TinDomain command.....	111	Tracking Analyst.....	189
TinEdge command.....	112	Tools	
TINHull. <i>See TINDomain command</i>		custom.....	34
TINLattice. <i>See TINRaster command</i>		model.....	97
TinLine command.....	112	script.....	46
TINLines. <i>See TINEdge command</i>		system.....	93
TinNode command.....	112	Toolset	
TinPolygonTag command.....	112	Aggregate.....	48
TinPolygonVolume command.....	127	Analysis.....	49
TinRaster command.....	113	Analyzing Patterns.....	67
TINs		Bitwise.....	165
calculating aspect.....	127	Caching.....	51
calculating contours.....	127	Composite Features.....	31
calculating slope.....	127	Conditional.....	139
converting to raster datasets.....	113	Conversion.....	17
creating.....	125	Database.....	56
editing.....	126	Data Comparison.....	52
extracting edges.....	112	Data Management.....	36
extracting interpolation zones.....	111	Density.....	140
extracting nodes.....	112	described.....	11
extracting polygon tag information.....	112	Disconnected Editing.....	39
extracting triangle polygons.....	113	Distance.....	141
TinSlope command.....	127	Distributed Geodatabase.....	39
TINSpot. <i>See SurfaceSpot command</i>		Domains.....	39
TIN Surface toolset.....	126	Extract.....	11
TinTriangle command.....	20	Extraction.....	145
To CAD toolset.....	19	Feature.....	69
To Coverage toolset.....	20	Feature Class.....	18
To dBASE toolset.....	42	Features.....	53
To Geodatabase toolset.....	42	Fields.....	57

From Coverage.....	8
From Feature Class	111
From File.....	111
From Raster.....	113
From Terrain.....	40
From TIN	98
Functional Surface	114
General.....	59
Generalization	149, 61, 70
Graphic Quality.....	151
Groundwater.....	67
Hydrology	154
Indexes	72
Interpolation	12
Items.....	72
Joins	157
Layers and Table Views	66
licensing	A-1
Local.....	159
Logical.....	13
Map Algebra.....	160
Mapping Clusters	160
Masking.....	41
Math	75
Measuring Geographic Distributions	99
Metadata.....	179
Multivariate.....	18
Neighborhood.....	19
Network Dataset.....	135
Overlay.....	75
Projections.....	134
Projections and Transformations.....	68
Proximity.....	...
Raster.....	70
Raster Catalog	76
Raster Creation.....	42
Raster Dataset.....	73
Raster Interpolation.....	22
Raster Math.....	24
Raster Processing	75
Raster Properties	77
Raster Reclass	120
Raster Surface	121
Reclass.....	182
Relationship Classes.....	78
Representation Management.....	183
Solar Radiation.....	42
Subtypes	79
Surface.....	77
Symbolization Refinement.....	78
Table.....	80
Tables	123
Terrain	124
TIN Creation	125
TIN Surface.....	126
To CAD	19

To Coverage	20
To dBASE	79
To Geodatabase	24
To KML.....	22
Tolerances	67
Topology	135
To Raster	23
To Shapefile.....	25
Trigonometric.....	173
Turn Feature Class	57
Versions	83
Topology	
removing rules.....	82
toolset	116
To Raster toolset	23
Transform command.....	117
Transforming	
route events	172
TransformRouteEvents command.....	135
TransposeTimeFields command	58
Trend command	118
Trigonometric toolset.....	173

U

UncompressFileGeodatabaseData command.....	58
Uncompressing	
dataset.....	58
Underpasses	
creating.....	7
UnregisterAsVersioned command	84
Unregistering	
as versioned.....	84
UpdateAnnotation command	52
UpdateByAlternateIDFields command.....	136
UpdateByGeometry command.....	136
Update command	13
UpdateDiagram command	137
UpdateDiagrams command.....	137
UpdateGlobeServerCache command	95
UpdateMapServerCache command	95
UpdateOverride command
Updating	
IDs	46
UpgradeSpatialReference command.....	47

V

ValidateTopology command	82
Variety. <i>See CellStatistics command</i>	
Vehicle route problem	
make layer	134
Velocity. <i>See DarcyVelocity command</i>	
Versions	
changing edit session.....	83
creating.....	83
deleting.....	83
reconciling.....	83
registering.....	83
unregistering.....	84
Viewshed command	186
VisdeCode. <i>See ObserverPoints command</i>	
Visibility. <i>See LineOfSight command. See Viewshed command</i>	
Volume. <i>See SurfaceVolume command</i>	
VPF	
creating tile topology.....	34
VPFExport command.....	43

W

Warp command	71
Watershed command	154
WCS	
create layer	67
WeightedOverlay command.....	180
WeightedSum command	180
Workspace	
creating.....	84
toolset	84
workspace cache	
clearing.....	
Workspace Management toolset	

X

XOr. <i>See Exclusive or. See Exclusive Or</i>	
XSLTransform command.....	19

Z

ZonalArea. <i>See TabulateArea command</i>	
ZonalCentroid. <i>See ZonalGeometry command</i>	
ZonalFill command.....	186
ZonalGeometryAsTable command	187
ZonalGeometry command	187. <i>See also ZonalGeometryAsTable command</i>
ZonalMajority. <i>See ZonalStatistics command</i>	
ZonalMax. <i>See ZonalStatistics command</i>	
ZonalMean. <i>See ZonalStatistics command</i>	
ZonalMedian. <i>See ZonalStatistics command</i>	
ZonalMin. <i>See ZonalStatistics command</i>	
ZonalMinority. <i>See ZonalStatistics command</i>	
ZonalPerimeter. <i>See ZonalGeometry command</i>	
ZonalRange. <i>See ZonalStatistics command</i>	
ZonalStatisticsAsTable command.....	187
ZonalStatistics command	187
ZonalStats. <i>See ZonalStatisticsAsTable command</i>	
ZonalStd. <i>See ZonalStatistics command</i>	
ZonalSum. <i>See ZonalStatistics command</i>	
ZonalThickness. <i>See ZonalGeometry command</i>	
Zonal toolset.....	186
ZonalVariety. <i>See ZonalStatistics command</i>	
ZRenderer command.....	104
Z score rendering	104
Z-value	
calculating	131

Appendix A: Tool 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	✓	✓	✓
Spatial Join	✓	✓	✓
Symmetrical Difference			✓
Union	✓	✓	✓
Update			✓
Proximity toolset			
Buffer	✓	✓	✓
Create Thiessen Polygons			✓
Generate Near Table			
Multiple Ring Buffer	✓	✓	✓
Near			✓
Point Distance			✓
Statistics toolset			
Frequency			✓
Summary Statistics	✓	✓	✓

Cartography toolbox

Toolset/Tool	ArcView	ArcEditor	ArcInfo
Graphic Quality toolset			
Detect Graphic Conflict			✓
Masking Tools toolset			
Cul-De-Sac Masks			✓
Feature Outline Masks			✓
Intersecting Layers Masks			✓
Representation Management toolset			
Add Representation			✓
Calculate Representation Rule			✓
Drop Representation			✓
Remove Override			✓
Select Feature By Override			✓
Set Layer Representation			✓
Update Override			✓
Symbolization Refinement toolset			
Align Marker To Stroke Or Fill			✓
Calculate Geodesic Angle			✓
Calculate Line Caps			✓
Calculate Polygon Main Angle			✓
Create Overpass			✓
Create Underpass			✓
Disperse Markers			✓
Set Representation Control Point By Intersect			✓
Set Representation Control Point By Angle			✓

Conversion toolbox

Toolset/Tool	ArcView	ArcEditor	ArcInfo
From Raster toolset			
Raster To ASCII	✓	✓	✓
Raster To Float	✓	✓	✓
Raster To Point	✓	✓	✓
Raster To Polygon	✓	✓	✓
Raster To Polyline	✓	✓	✓
From WFS			
WFS To Feature Class		✓	✓

Metadata toolset (Conversion toolbox cont.)	ArcView	ArcEditor	ArcInfo
ESRI Metadata Translator	✓	✓	✓
Metadata Importer	✓	✓	✓
Metadata Publisher	✓	✓	✓
USGS MP Metadata Translator	✓	✓	✓
XSLT Translator	✓	✓	✓
To CAD toolset			
Add CAD Fields			✓
Create CAD XData			✓
Export To CAD			✓
Set CAD Alias			✓
To Coverage toolset			
Feature Class To Coverage			✓
To dBASE toolset			
Table To dBASE (multiple)	✓	✓	✓
To Geodatabase toolset			
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 KML			
Layer To KML	✓	✓	✓
Map To KML	✓	✓	✓
To Raster toolset			
ASCII To Raster	✓	✓	✓
DEM To Raster	✓	✓	✓
Feature To Raster	✓	✓	✓
Float To Raster	✓	✓	✓
Point To Raster	✓	✓	✓
Polygon To Raster	✓	✓	✓
Polyline To Raster	✓	✓	✓
Raster To Other Format (multiple)	✓	✓	✓
To Shapefile toolset			
Feature Class To Shapefile (multiple)	✓	✓	✓

Data Management toolbox

Toolset/Tool	ArcView	ArcEditor	ArcInfo
Data Comparison toolset			
Feature Compare	✓	✓	✓
File Compare	✓	✓	✓
Raster Compare	✓	✓	✓
Table Compare	✓	✓	✓
TIN Compare	✓	✓	✓
Database toolset			
Clear Workspace Cache	✓	✓	✓
Compact	✓	✓	✓
Compress	✓	✓	✓
Migrate Storage			✓
Upgrade Spatial Reference	✓	✓	✓
Disconnected Editing toolset			
Check In		✓	✓
Check In From Delta		✓	✓
Check Out		✓	✓
Export To Delta		✓	✓
Distributed Geodatabase toolset			
Add Global IDs			✓
Compare Replica Schema			✓
Create Replica			✓
Create Replica Footprints			✓
Create Replica From Server			✓
Export Acknowledgment Message			✓
Export Data Change Message			✓
Export Replica Schema			✓
Import Message			✓
Import Replica Schema			✓
Re-Export Unacknowledged Messages			✓
Synchronize Changes			✓
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 (Data Management cont.)	ArcView	ArcEditor	ArcInfo
Append Annotation Feature Classes	✓	✓	✓
Calculate Default Cluster Tolerance	✓	✓	✓
Calculate Default Spatial Grid Index	✓	✓	✓
Create Feature Class	✓	✓	✓
Create Fishnet	✓	✓	✓
Create Random Points	✓	✓	✓
Integrate	✓	✓	✓
Update Annotation Feature Class	✓	✓	✓
Features toolset			
Add XY Coordinates	✓	✓	✓
Adjust 3D Z	✓	✓	✓
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			
Add Field	✓	✓	✓
Assign Default To Field	✓	✓	✓
Calculate End Date	✓	✓	✓
Calculate Field	✓	✓	✓
Delete Field	✓	✓	✓
Transpose Time Fields	✓	✓	✓
File Geodatabase toolset			
Compress File Geodatabase Data	✓	✓	✓
Uncompress File Geodatabase Data	✓	✓	✓
General toolset			
Append	✓	✓	✓
Calculate Value	✓	✓	✓
Copy	✓	✓	✓
Delete	✓	✓	✓
Merge	✓	✓	✓
Merge Branch	✓	✓	✓
Rename	✓	✓	✓
Select Data	✓	✓	✓

Generalization toolset (Data Management cont.)	ArcView	ArcEditor	ArcInfo
Aggregate Polygons			✓
Collapse Dual Lines To Centerline			✓
Dissolve	✓	✓	✓
Eliminate			✓
Simplify Building			✓
Simplify Line		✓	✓
Simplify Polygon			✓
Smooth Line		✓	✓
Smooth Polygon			✓
Indexes toolset			
Add Attribute Index	✓	✓	✓
Add Spatial Index	✓	✓	✓
Remove Attribute Index	✓	✓	✓
Remove Spatial Index	✓	✓	✓
Joins toolset			
Add Join	✓	✓	✓
Join Field			✓
Remove Join	✓	✓	✓
Layers and Table Views toolset			
Apply Symbology From Layer	✓	✓	✓
Make Feature Layer	✓	✓	✓
Make Image Server Layer	✓	✓	✓
Make Query Table	✓	✓	✓
Make Raster Catalog Layer	✓	✓	✓
Make Raster Layer	✓	✓	✓
Make Table View	✓	✓	✓
Make WCS Layer	✓	✓	✓
Make XY Event Layer	✓	✓	✓
Save To Layer File	✓	✓	✓
Select Layer By Attribute	✓	✓	✓
Select Layer By Location	✓	✓	✓
Projections and Transformations toolset			
Create Custom Geographic Transformation	✓	✓	✓
Define Projection	✓	✓	✓
Feature toolset			
Batch Project	✓	✓	✓
Create Spatial Reference	✓	✓	✓
Project	✓	✓	✓

Raster toolset (Data Management cont.)	ArcView	ArcEditor	ArcInfo
Flip	✓	✓	✓
Mirror	✓	✓	✓
Project Raster	✓	✓	✓
Rescale	✓	✓	✓
Rotate	✓	✓	✓
Shift	✓	✓	✓
Warp	✓	✓	✓
Raster toolset			
Raster Catalog			
Copy Raster Catalog Items	✓	✓	✓
Create Raster Catalog	✓	✓	✓
Delete Raster Catalog Items	✓	✓	✓
Export Raster Catalog Paths	✓	✓	✓
Repair Raster Catalog Paths	✓	✓	✓
Workspace To Raster Catalog	✓	✓	✓
Raster Dataset			
Copy Raster	✓	✓	✓
Create Random Raster	✓	✓	✓
Create Raster Dataset	✓	✓	✓
Mosaic	✓	✓	✓
Mosaic To New Raster	✓	✓	✓
Raster Catalog To Raster Dataset	✓	✓	✓
Workspace To Raster Dataset	✓	✓	✓
Raster Processing			
Clip	✓	✓	✓
Composite Bands	✓	✓	✓
Create Ortho-Corrected Raster Dataset	✓	✓	✓
Create Pan-Sharpened Raster Dataset	✓	✓	✓
Extract Subdataset	✓	✓	✓
Resample	✓	✓	✓
Raster Properties			
Add Colormap	✓	✓	✓
Batch Build Pyramids	✓	✓	✓
Batch Calculate Statistics	✓	✓	✓
Build Pyramids	✓	✓	✓
Build Raster Attribute Table	✓	✓	✓
Calculate Statistics	✓	✓	✓
Delete Colormap	✓	✓	✓
Delete Raster Attribute Table	✓	✓	✓
Export Raster World File	✓	✓	✓
Get Cell Value	✓	✓	✓
Get Raster Properties	✓	✓	✓

Relationship Classes toolset (Data Mgmt.)	ArcView	ArcEditor	ArcInfo
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			
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 File Geodatabase	✓	✓	✓
Create Folder	✓	✓	✓
Create Personal Geodatabase	✓	✓	✓

Geocoding toolbox

Tool	ArcView	ArcEditor	ArcInfo
Create Address Locator	✓	✓	✓
Geocode Addresses	✓	✓	✓
Rebuild Address Locator	✓	✓	✓
Rematch Addresses	✓	✓	✓
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	✓	✓	✓

Mobile Toolbox

Tool	ArcView	ArcEditor	ArcInfo
Create Mobile Basemap	✓	✓	✓
Generate Mobile Cache	✓	✓	✓

Multidimension Toolbox

Tool	ArcView	ArcEditor	ArcInfo
Feature To NetCDF	✓	✓	✓
Make NetCDF Feature Layer	✓	✓	✓
Make NetCDF Raster Layer	✓	✓	✓
Make NetCDF Table View	✓	✓	✓
Raster To NetCDF	✓	✓	✓
Select By Dimension	✓	✓	✓
Table To NetCDF	✓	✓	✓

Server Toolbox

Tool	ArcView	ArcEditor	ArcInfo
Caching toolset			
Create Map Server Cache	✓	✓	✓
Delete Globe Server Cache	✓*	✓*	✓*
Delete Map Server Cache	✓	✓	✓
Generate Map Server Cache Tiling Scheme	✓	✓	✓
Manage Globe Server Cache Tiles	✓*	✓*	✓*
Manage Map Server Cache Scales	✓	✓	✓
Manage Map Server Cache Tiles	✓	✓	✓

* Requires 3D Analyst extension

Spatial Statistics toolbox

Toolset/Tool	ArcView	ArcEditor	ArcInfo
Analyzing Patterns toolset			
Average Nearest Neighbor	✓	✓	✓
High/Low Clustering (Getis-Ord General G)	✓	✓	✓
Multi-Distance Spatial Cluster Analysis (Ripley's K Function)	✓	✓	✓
Spatial Autocorrelation (Morans I)	✓	✓	✓
Mapping Clusters toolset			
Cluster And Outlier Analysis (Anselin Local Morans I)	✓	✓	✓
Hot Spot Analysis (Getis-Ord Gi*)	✓	✓	✓
Measuring Geographic Distributions toolset			
Central Feature	✓	✓	✓
Directional Distribution (Standard Deviation Ellipse)	✓	✓	✓
Linear Directional Mean	✓	✓	✓
Mean Center	✓	✓	✓
Standard Distance	✓	✓	
Modeling Spatial Relationships			
Generate Network Spatial Weights	✓**	✓**	✓**
Generate Spatial Weights Matrix	✓	✓	✓
Geographically Weighted Regression	✓	✓	✓
Ordinary Least Squares	***	***	✓

** Requires the Network Analyst extension

*** Requires the Spatial Analyst extension

Rendering toolset (Spatial Statistics cont.)	ArcView	ArcEditor	ArcInfo
Cluster/Outlier Analysis With Rendering	✓	✓	✓
Collect Events With Rendering	✓	✓	✓
Count Rendering	✓	✓	✓
Hot Spot Analysis With Rendering	✓	✓	✓
Z Score Rendering	✓	✓	✓
Utilities toolset			
Calculate Areas	✓	✓	✓
Calculate Distance Band From Neighbor Count	✓	✓	✓
Collect Events	✓	✓	✓
Convert Spatial Weights Matrix To Table	✓	✓	✓
Export Feature Attribute To ASCII	✓	✓	✓

