



# ARCPY & RASTER PROCESSING IN ARCGIS 10

---

Best Practices

# IMAGERY

---

# New Raster Additions & Improvements

- **Faster graphics rendering**  
Graphics hardware acceleration option
- **Better raster format support**  
Read and writes more raster formats than previous versions
- **Image Analysis Window**  
On-the-fly image processing
- **Mosaic Dataset**  
Foundation for image server

# Raster Formats

- ArcGIS 10 supports 69 different raster types...
  - ArcGIS 9.3.1 supported 37 raster types
- All supported formats are made possible through the Geospatial Data Abstraction Library (GDAL).
- New format examples...
  - Hierarchical Data Format (HDF) 5
  - Terragen Terrain
  - Golden Software Grid (.grd)
  - Magellan Mapsend
  - GRIB

# Image Compression

- New compression types for TIFF file format.
  - Packbits (Lossless)
  - CCITT Group 3 (Lossless)
  - CCITT Group 4 (Lossless)
  - CCITT (1D) (Lossless)
- Resources
  - <http://en.wikipedia.org/wiki/Packbits>
  - [http://www.fileformat.info/mirror/egff/ch09\\_05.htm](http://www.fileformat.info/mirror/egff/ch09_05.htm)



# Image Analysis Window

- On-the-fly image processing
  - Generates temporary rasters
- Manual display controls
  - Contrast    Brightness    Transparency    Gamma
- Other display controls
  - Zoom to Raster Resolution    Swipe    Flicker Layer
- Automatic Processing
  - Clip    Mask    Orthorectify    Pan sharpening
  - Shaded Relief    Mosaic    NDVI    Composite Bands

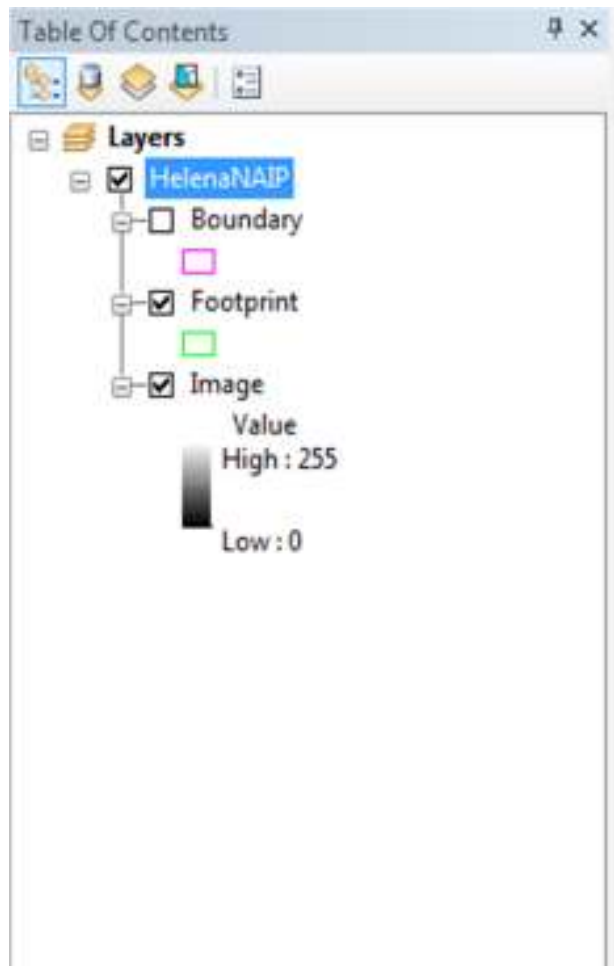
# Mosaic Raster Dataset

- Created within a geodatabase.
- No data is loaded. Acts as a pointer to a workspace/raster type.
- Processing done on-the-fly. Original images are unaltered.
- Includes additional querying capabilities as well as the new processing functions.
- Does not have to contain continuous images. Can be disjointed sets of images with varying resolutions.
- Easy to share via ArcGIS Server – Image Server Extension.
- Addition of Overviews speeds up loading times in ArcGIS Desktop and Server.

# Referenced Mosaic Raster Dataset

- Can be created inside or outside of a geodatabase...
  - No overviews are built
  - No boundaries are created
  - Different file extension (.amd)
- Read only
  - You cannot add rasters to a reference mosaic dataset
- Provides another means of distributing rasters.
  - Examples: Distributing certain rasters to users from a Mosaic Dataset
- Can also be used with raster catalogs as well.

# Elements of the Mosaic Raster Dataset



- **Boundary Layer**
  - Outer extent of the entire raster defined by the footprint(s).
- **Footprint Layer**
  - Displays the extent of each individual raster referenced in the Mosaic Raster Dataset.
- **Image Layer**
  - Equivalent to a raster layer.
  - Controls “the rendering of the dynamically mosaicked image” (ESRI).



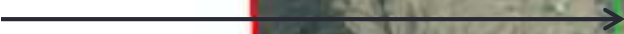
Boundary Layer



Footprint Layer



Image Layer



# Raster Mosaic Dataset Metadata

- When loading rasters by file type each raster type is added based on it's associated metadata...
  - Landsat 7 ETM+: .MET - .FST – MTL.TXT – WO.TXT
- Examples of metadata added...
  - MinPS – MaxPS – Acquisition Date – Cloud Cover – Sun Azimuth  
Sun Elevation – Sensor Name etc.
- Metadata **has to match** the standards of those provided by the imagery vendor or the metadata and imagery will fail to load...
  - Example: Raster Mosaic Dataset will not accept Landsat imagery from MontanaView.

# Raster Mosaic Datasets & Attribute Tables

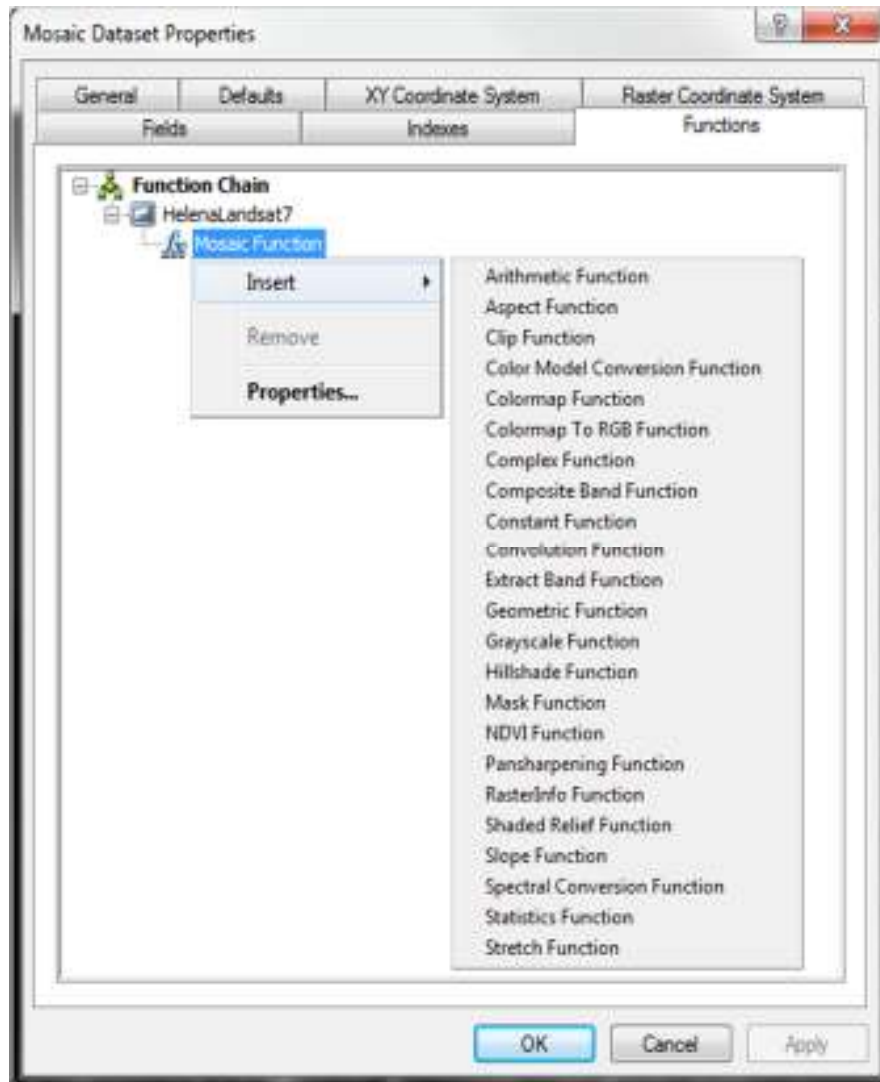
- When you add rasters with attribute tables, the attribute tables **are not** loaded.
- A mosaic dataset attribute table is loaded with the following default fields...
  - ObjectID
  - Raster
  - Name
  - MinPS
  - MaxPS
  - LowPS
  - HighPS
  - Category
  - Tag
  - GroupName
  - CenterX
  - CenterY
  - ZOrder
  - SOrder



# Mosaic Dataset Color Correction

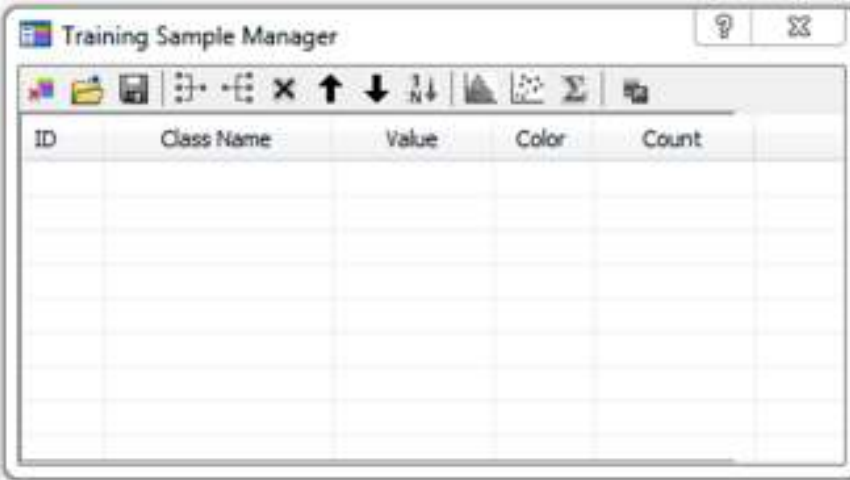
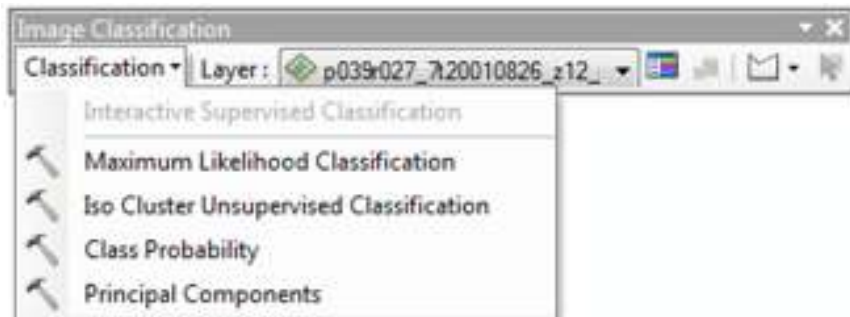
- Accessed through the customize menu.
- Mosaic wide (seamless) color correction is done using the Color Balancing option.
- Color Correction requires statistics for each band.
- Colors can be excluded using the Excluded Area or Percentage option.

# Raster with Functions



- Functions applied to one or more mosaic raster datasets
- Processed entirely on-the-fly.
- Ability to add multiple functions to a single mosaic dataset
- Changes become permanent when you export the raster as a layer file or a supported raster dataset.

# Improved Image Classification



- Unsupervised and Supervised Image Classification
- Training Sample Manager to aid in analyzing training samples
  - Histograms
  - Scatterplots
  - Statistics
- Unsupervised classifications can create a signature file to use in the Maximum Likelihood Classification

# RASTER GEOPROCESSING TOOLS

---

# Geoprocessing

- Most of the new raster geoprocessing tools belong to the Raster Mosaic Dataset.
- Other new raster geoprocessing include...
  - Split Raster
  - Build Pyramids & Statistics
  - Make Mosaic Layer
  - Raster to DTED
- Improved tools...
  - Mosaic to New Raster
  - Add Colormap

# Split Raster

- Split rasters in the X & Y direction via two methods...
  - Size of Tile (based on pixels or mapping units)
  - Number of Tiles
- Does not split attribute tables.
- Various output formats...
  - TIFF
  - ENVI
  - GRID
  - Imagine
  - JPEG
  - PNG

# Build Pyramids & Statistics

- Batch tool for building pyramids and statistics of all rasters in a workspace
- Can also scan rasters in a subdirectory
- All raster formats are supported
- In Arc10 build pyramids is enabled by default when rasters are loaded. This is the recommended Arc10 workflow.

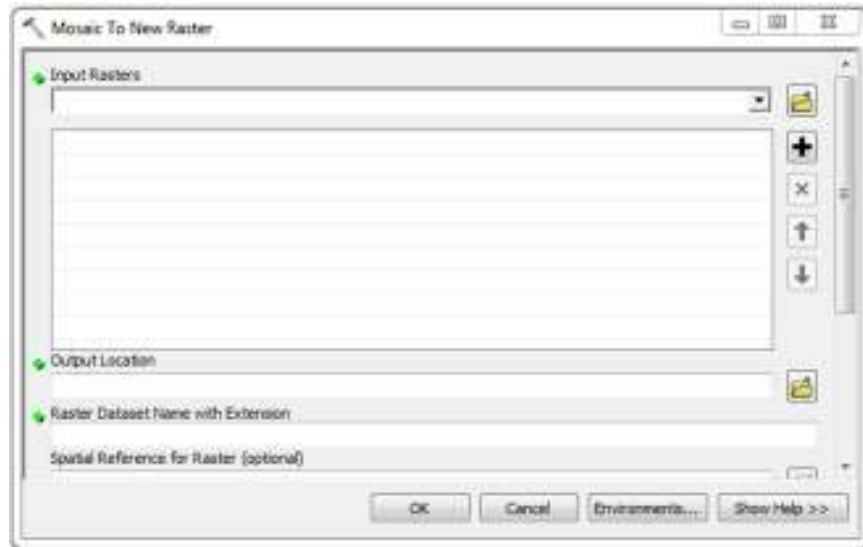
# Make Mosaic Layer

- Creates a temporary raster layer
- Used in conjunction with a mosaic dataset
- “This tool can be used to make a temporary layer, so you can work with a specified subset of bands within a mosaic dataset.” (ESRI)
- To make the layer permanent “Save as Layer”

# Raster to DTED

- DTED > Digital Terrain Elevation Data (NGA developed)
- Primarily used in the Military for various purposes
- Input raster will be a single band raster with elevation data
- Three different levels
  - DTED Level 0 (1 kilometer)
  - DTED Level 1 (~100 meters)
  - DTED Level 2 (30 meters)
- Nationwide DTED Level 0 is available for public use

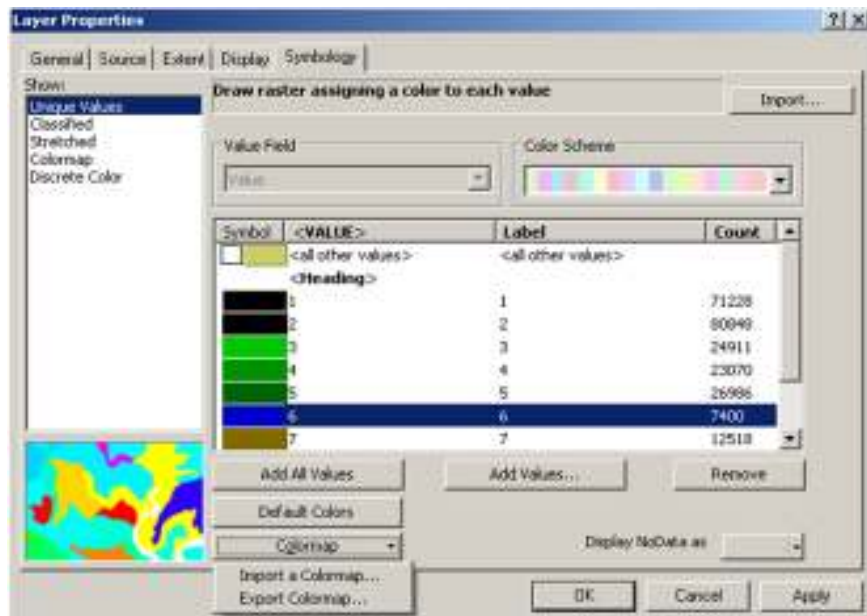
# Mosaic to New Raster & Add Colormap



- Mosaic to New Raster
  - No longer a script but now a dedicated system tool.

- Add Colormap

- Create your own custom color maps within the Unique Values Symbology and export it.



- Add Colormap geoprocessing tool allows you to apply your newly created color scheme to your rasters.

# DEMO

---

Image Analysis Window

Raster Mosaic Dataset

# QUESTION AND ANSWER SESSION

---

# PYTHON

---

An introduction to the new ArcPy Site Package including  
Python 2.6

# INTRO TO ARCPY

---

# ArcPy

- The new ArcPy Site Package
  - Fits Python Framework
- Builds on the arcgisscripting module
- Begin all scripts with **import arcpy**
- Difference between 9.3.1 & 10
  - Replace gp. with arcpy.
    - gp.CopyFeatures\_management(input, output)
    - arcpy.CopyFeatures\_management(input, output)
- >>> help(arcpy)

# Python Framework in ArcGIS 10

Term	Definition
ArcPy	ArcGIS 10 introduces ArcPy (often referred to as the ArcPy site-package), which provides Python access for all geoprocessing tools, including extensions, as well as a wide variety of useful functions and classes for working with and interrogating GIS data. A site-package is Python's term for a library that adds additional functions to Python. Using Python and ArcPy, you can develop an infinite number of useful programs that operate on geographic data.
ArcPy modules	A module is a python file that generally includes functions and classes. ArcPy is supported by a series of modules, including a mapping module ( <b>arcpy.mapping</b> ), a Spatial Analyst module ( <b>arcpy.sa</b> ), and a Geostatistical Analyst module ( <b>arcpy.ga</b> ).
ArcPy classes	A class is analogous to an architectural blueprint. The blueprint provides the framework for how to create something. Classes can be used to create objects, often referred to as an instance. ArcPy classes, such as the SpatialReference and Extent classes, are often used as shortcuts to complete geoprocessing tool parameters that would otherwise have a more complicated string equivalent.
ArcPy functions	A function is a defined bit of functionality that does a specific task and can be incorporated into a larger program. In ArcPy all geoprocessing tools are provided as functions, but not all functions are geoprocessing tools. In addition to tools, ArcPy provides a number of functions to better support geoprocessing Python workflows. Functions or methods can be used to list certain datasets, retrieve a dataset's properties, validate a table name before adding it to a geodatabase, or perform many other useful scripting tasks.

Source: ESRI

# Python Framework in ArcGIS 10

- Examples
  - Modules
    - Spatial Analyst
    - Mapping
    - Geostatistical
  - Classes
    - Extent
    - Point
  - Function
    - Search Cursor
    - ListRasters
    - Describe

# Python 2.6

- New Modules
  - Multiprocessing Module
    - Allows for spawning multiple processes
    - Sidesteps Global Interpreter Lock
    - Important for HPC
- Python 2.6 Updates
  - 259 Patches
  - 612 Bugs Fixed

Source: What's New in Python 2.6

# ArcPy

- Once ArcPy is imported you have access to all of the standard toolboxes
  - Analysis
  - Cartography
  - Conversion
  - Data Management
  - Editing
  - Geocoding
  - Linear Referencing
  - Multidimension
  - Spatial Statistics

# MODULES

---

# ArcPy

- Similar to other Python packages you have modules
- ArcPy Modules
  - Mapping Module
  - Spatial Analyst Module
  - Geostatistical Analyst Module

# Mapping Module

- Open Map Documents and Layers
- Alter the Contents of the Document or Layer
- Print the Modified Document
- Export the Modified Document
- Save the Modified Document
- Module uses Functions and Classes
- Example:

```
import arcpy  
mxd = arcpy.mapping.MapDocument('c:/pnw/annualdischarge.mxd')  
arcpy.mapping.ExportToJPEG(mxd,'c:/temp/annualdischarge.jpg')  
del mxd
```

# Geostatistical Analyst

- Used for defining parameters for Geostatistical Analyst Tools
- Uses classes
- Example:

```
import arcpy  
result = arcpy.CrossValidation_ga('c:/rap/dem.lyr')  
print str(cvResult.rootMeanSquare)
```

# Spatial Analyst Module

- Used for defining parameters in Spatial Analyst Tools
- Uses Classes, Operators, and Functions
- Raster is a class, add is a operator
- Example:

```
import arcpy
from arcpy import env
from arcpy.sa import *
env.workspace = 'c:/temp'
outplus = Raster('temp98') + Raster('temp99')
```

# ArcPy Environment Settings

- Additional parameters used for input into tools
- Separate from tools
  - Set workspace environment
  - Get current raster cell size
- Example:

```
import arcpy  
arcpy.env.workspace('c:/geoshare/thebestgdbbever.gdb')
```

# FUNCTIONALITY

---

# New ArcPy Syntax

- The For Loop replaces the While Loop for listing
- Syntax Change for the Search Cursors
  - Example

```
import arcpy
rows = arcpy.SearchCursor('c:/geodatabase/RAP.shp','','',
'Name;WS_Area;WS_Z_MEAN')
for row in rows:
    print row.Name
```

# Old Arc 9.3.1 Syntax vs. New Arc 10 Syntax

## New Arc 10 Syntax

```
for rasters in arcpy.ListRasters():  
    print rasters
```

For Loop

## Old Arc 9.3 Syntax

```
RasterList = gp.ListRasters():  
Rasters = RasterList.next()  
while Rasters:  
    print Rasters  
    Rasters = RasterList.next()
```

While Loop

# New ArcPy Functionality

- NumPyArraytoRaster & RastertoNumpyArray
  - Simple integration of HDF5,HDF4, and Binary raster data
  - Allows for simple and complex raster analysis using just the numerical array (NumPy and SciPy)
- PointstoLine
  - Creates polylines from points
- ExportMetadata
  - Exports Metadata to XML via Translator (FGDC, ISO 19139)
- Much Much More!!

# Spatial Reference Class

- No need to embed the entire projection information into Python Code
- Spatial Reference Class

```
import arcpy
```

```
prjfile = "c:/Program Files/ArcGIS/Desktop10.0/Coordinate Systems/Projected  
Coordinate Systems/Continental/North America/USA Contiguous Equidistant  
Conic.prj"
```

```
spatialreference = arcpy.SpatialReference(prjFile)
```

# Calculate Field

- New Python Syntax-Math.sin()
- Uses Math, String, Datetime Python Modules
- Use the !HydroID!

`arcpy.CalculateField_management("kamfloodplains","linelen","math.sin(!linelen!)", "PYTHON")`



# Geometry operators

- Python geometry objects support relational operators
  - Contains
  - Crosses
  - Disjoint
  - Equals
  - Overlaps
  - Touches
  - Within

```
from arcpy import *
```

```
p1 = Polygon(Array([[Point(1,5),Point(2,4),Point(6,5)]]))
```

```
p2 = Polygon(Array([[Point(3,5),Point(6,4),Point(8,5)]]))
```

```
p1.overlaps(p2)
```

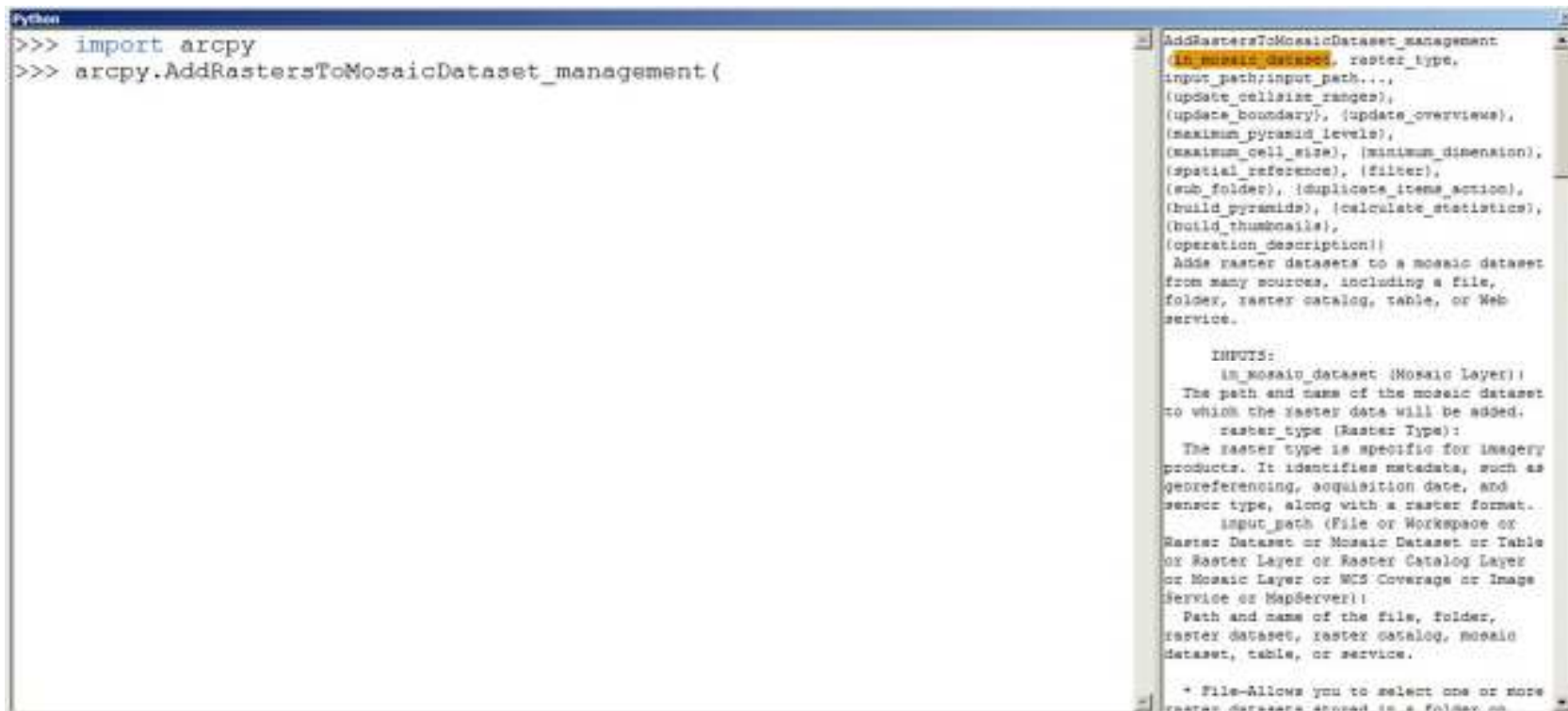
```
True
```

# Useful Python Packages

- Scipy
  - Supplements Numpy with high level science modules
- RPy or RPy2
  - Python connection to R
- Psycopg (SDE)
  - PostgreSQL database adapter for Python
  - Helpful for PostgreSQL ArcSDE
- Matplotlib
  - 2D plotting library
- Pysal
  - Read DBF Files
- MUST USE Python 2.6 package!!!

# Python Window

- All Python functionality right within ArcMap
- No need to write Python Scripts
- Can also incorporate other Python Packages
- Intellisense for ArcPy and other Python Packages



```
>>> import arcpy
>>> arcpy.AddRastersToMosaicDataset_management(
```

```
AddRastersToMosaicDataset_management
in_mosaic_dataset, raster_type,
input_path:input_path...
(update_cellsize_ranges),
(update_boundary), (update_overviews),
(maximum_pyramid_levels),
(maximum_cell_size), (minimum_dimension),
(spatial_reference), (filter),
(sub_folder), (duplicates_items_action),
(build_pyramids), (calculate_statistics),
(build_thumbnails),
(operation_description)
Add raster datasets to a mosaic dataset
from many sources, including a file,
folder, raster catalog, table, or Web
service.

INPUTS:
in_mosaic_dataset (Mosaic Layer):
The path and name of the mosaic dataset
to which the raster data will be added.
raster_type (Raster Type):
The raster type is specific for imagery
products. It identifies metadata, such as
georeferencing, acquisition date, and
sensor type, along with a raster format.
input_path (File or Workspace or
Raster Dataset or Mosaic Dataset or Table
or Raster Layer or Raster Catalog Layer
or Mosaic Layer or WCS Coverage or Image
Service or MapServer):
Path and name of the file, folder,
raster dataset, raster catalog, mosaic
dataset, table, or service.

* File-Allows you to select one or more
raster datasets stored in a folder or
```

# RASTER PROCESSING IN ARCPY

---

# The Spatial Analyst Module

- Classes
  - Used for defining parameters for Spatial Analyst Tools
  - Remap Range - **A list identifying what ranges of input values should be reclassified to in an output raster.**
- Operators (Map Algebra)
  - Arithmetic
  - Bitwise
  - Boolean
  - Relational
  - Addition – **Adding two rasters together**
- Functions
  - Apply Environment - **Creates a new raster that is a copy of the input raster with the current environment settings being applied.**

# Spatial Analyst

- Proper Import Syntax
  - `from arcpy.sa import *`
  - This syntax allows all module contents to be imported
  - No need to add syntax for the toolbox

- Example

```
from arcpy.sa import *  
Plus("dem", 54)
```

# Map Algebra

- Map Algebra has been integrated into Python.
- Can be accessed through 3 environments
  - Raster Calculator
  - Python Window
  - Python Integrated Development Environment (IDE)
- More complex Map Algebra expressions are better suited for either the Python Window in ArcMap or a Python IDE (such as PythonWin or IDLE).



- Raster Calculator performs the same as previous versions.
- No longer necessary to specify output raster in expression.
  - Example: raster = Expression
- Tool variable available to specify the output raster and type.

# NumPy - Raster to Numpy Array

- Useful for raster computation
- Why? Computers are meant for number crunching
- Possibilities – NDVI, Resample, Selection
- Important Factors for NumPy Integration
  - Lower Left Corner of the Raster
  - The Number of Rows
  - The Number of Columns
  - No Data Value
- Limits: Has Array Size Limits based on RAM
  - Alaska DEM 39383 Rows, 49563 Column
  - 4GB

# NumPy - RasterToNumPyArray

```
>>> import arcpy
>>> numpyarray = arcpy.RasterToNumPyArray("dem")
>>> print numpyarray
[[ 964.72998047  967.79998779  968.91998291 ..., 836.15002441
  834.67999268  834.40997314]
 [ 957.90002441  964.7800293   966.84002686 ..., 836.80999756
  835.94000244  835.79998779]
 [ 953.41998291  962.05999756  963.98999023 ..., 835.48999023
  835.9699707   837.89001465]
 ...,
 [ 1316.64001465  1319.41003418  1321.10998535 ..., 1320.38000488
  1320.20996094  1318.95996094]
 [ 1316.72998047  1319.         1321.02001953 ..., 1322.22998047
  1322.88000488  1321.38000488]
 [ 1316.58996582  1318.95996094  1321.02001953 ..., 1323.11999512
  1323.39001465  1322.43994141]]
>>> print numpyarray.shape
(1858, 1269)

>>> from arcpy.sa import *

>>> finalextent = arcpy.Describe("dem").Extent
>>> print finalextent
599910 305550 637980 361290 NaN NaN NaN NaN

>>>
```

# NumPy - NumpyArraytoRaster

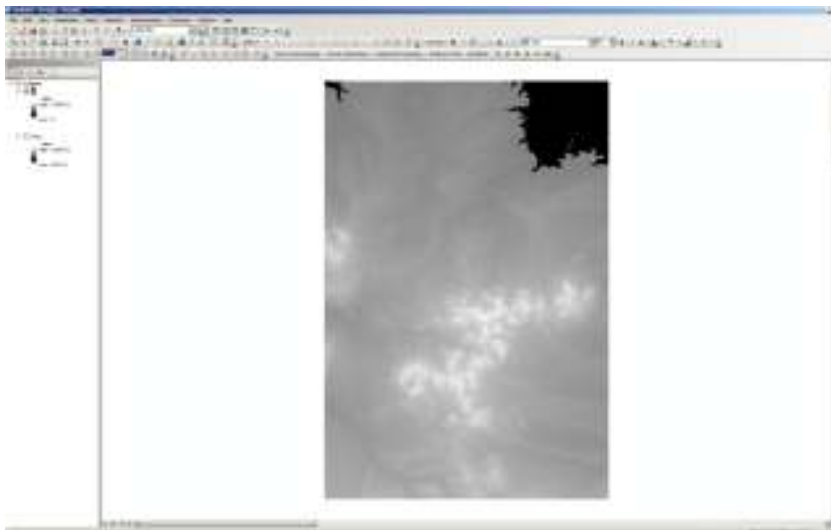
```
>>> import numpy
>>> final = dem[numpy.where(dem<1000)]=0
>>> print dem
[[ 0. 0. ... 0. 0.
  0. ] 0. 0. ... 0. 0.
  0. ] 0. 0. ... 0. 0.
  0. ]
...
 [ 1316.64001465 1319.41003418 1321.10998535 ..., 1320.38000488
   1320.20996094 1318.95996094]
 [ 1316.72998047 1319. 1321.02001953 ..., 1322.22998047
   1322.88000488 1321.38000488]
 [ 1316.58996582 1318.95996094 1321.02001953 ..., 1323.11999512
   1323.39001465 1322.43994141]]
>>> print dem.shape
(1858, 1269)

>>> newpoint = arcpy.Point(305550,599910)
>>> r = arcpy.NumPyArrayToRaster(dem,newpoint,30,30,-3.40282346639e+038)
>>>
```

# NumPy - NumpyArraytoRaster



Initial Raster



Altered Raster

# DEMO

---

ArcPy Demo

Python Window

# QUESTION AND ANSWER SESSION

---



THANKS!!!!

- If you would like a copy of this presentation please visit <http://rap.ntsg.umt.edu>