Misc Stuff

Open Source RS/GIS Python
Week 7

GDAL & OGR utilities

- Installed with FWTools and available from inside the FWTools Shell
- Many more GDAL utilities than OGR
- Full documentation available at
  - http://www.gdal.org/ogr/ogr_utilities.html
  - http://www.gdal.org/gdal_utilities.html
  - I'm only showing some of the options!
- Type the name of the utility in the shell to get the syntax

ogrinfo

- Provides info about a vector data set

```
Z:\data>ogrinfo
Usage: ogrinfo [--help-general] [-ro] [-q] [-where restricted_where]
               [-spat xmin ymin xmax ymax] [-fid fid]
               [-sql statement] [-al] [-so] [-fields={YES/NO}]
               [-geom={YES/NO/SUMMARY}][--formats]
               datasource_name [layer [layer ...]]
```

- If just pass name of shapefile it tells you the available layers

```
Z:\data>ogrinfo sites.shp
INFO: Open of `sites.shp'
      using driver `ESRI Shapefile' successful.
1: sites (Point)
```

- If you give it the name of the shapefile and the layer (or –al), it lists all features, their attributes, and their WKT geometries

```
Z:\data>ogrinfo -al sites.shp
INFO: Open of `sites.shp'
      using driver `ESRI Shapefile' successful.

Layer name: sites
Geometry: Point
Feature Count: 42
Extent: (428117.132465, 4591699.896760) - (491429.330686,
   4653007.208704)
Layer SRS WKT:
PROJCS["WGS_1984_UTM_Zone_12N",
    GEOGCS["GCS_WGS_1984",
        DATUM["WGS_1984",
            SPHEROID["WGS_1984",6378137.0,298.257223563]],
```

```
PRIMEM["Greenwich",0.0],
        UNIT["Degree",0.0174532925199433]],
    PROJECTION["Transverse_Mercator"],
    PARAMETER["False_Easting",500000.0],
    PARAMETER["False_Northing",0.0],
    PARAMETER["Central_Meridian",-111.0],
    PARAMETER["Scale_Factor",0.9996],
    PARAMETER["Latitude_Of_Origin",0.0],
    UNIT["Meter",1.0]]
ID: Integer (8.0)
COVER: String (20.0)
OGRFeature(sites):0
  ID (Integer) = 1
  COVER (String) = shrubs
  POINT (455552.41836086405 4641822.0536848791)

OGRFeature(sites):1
  ID (Integer) = 2
  COVER (String) = trees
  POINT (449959.84085133421 4633802.5085768746)
```

- Can get the stuff from the last example, except the features, with –so

```
ogrinfo –al –so sites.shp
```

- Can get just the feature info with –q

```
ogrinfo –al –q sites.shp
```

- Can use an attribute query with –where (use double quotes, at least on Windows)

```
Z:\data>ogrinfo -al -q -where "id<5" sites.shp
Layer name: sites
OGRFeature(sites):0
  ID (Integer) = 1
  COVER (String) = shrubs
  POINT (455552.41836086405 4641822.0536848791)
OGRFeature(sites):1
  ID (Integer) = 2
  COVER (String) = trees
  POINT (449959.84085133421 4633802.5085768746)
OGRFeature(sites):2
  ID (Integer) = 3
  COVER (String) = rocks
  POINT (441201.65343074972 4619029.6623252863)
OGRFeature(sites):3
  ID (Integer) = 4
  COVER (String) = grass
  POINT (468214.85800508264 4608688.6699491739)
```

- Can limit the spatial extent with –spat

```
Z:\data>ogrinfo -al -q -spat 46000 4590000 490000
   4600000 sites.shp

Layer name: sites
OGRFeature(sites):16
  ID (Integer) = 17
  COVER (String) = grass
  POINT (467898.29701397714 4597292.4742693771)

OGRFeature(sites):17
  ID (Integer) = 18
  COVER (String) = bare
  POINT (473068.79320203303 4592966.1407242697)

OGRFeature(sites):18
  ID (Integer) = 19
  COVER (String) = trees
  POINT (479716.57401524781 4591699.8967598472)
...
```

ogr2ogr

- Converts between vector formats

```
Z:\data>ogr2ogr
Usage: ogr2ogr [--help-general] [-skipfailures] [-append] [-update] [-gt n]
               [-select field_list] [-where restricted_where]
               [-sql <sql statement>]
               [-spat xmin ymin xmax ymax] [-preserve_fid] [-fid FID]
               [-a_srs srs_def] [-t_srs srs_def] [-s_srs srs_def]
               [-f format_name] [-overwrite] [[-dsco NAME=VALUE] ...]
               [-segmentize max_dist]
               dst_datasource_name src_datasource_name
               [-lco NAME=VALUE] [-nln name] [-nlt type] [layer [layer ...]]
 -f format_name: output file format name, possible values are:
     -f "ESRI Shapefile"
     -f "MapInfo File"
     -f "TIGER"
     -f "S57"
     -f "DGN"
     -f "Memory"
     -f "BNA"
     -f "CSV"
     -f "GML"
    ...
```

- Reproject a shapefile
  - **`-f "ESRI Shapefile"`**: output will be a shapefile
  - **`-t_srs "EPSG:4269"`**: output will be projected to EPSG 4269 (geographic NAD83)
  - Could use **`-s_srs "EPSG:????"`** if the input shapefile didn't have a .prj file
  - **`cache_towns_geo.shp`**: output filename
  - **`cache_towns.shp`**: input filename

```
ogr2ogr -f "ESRI Shapefile" -t_srs
  "EPSG:4269" cache_towns_geo.shp
  cache_towns.shp
```

- Reproject a shapefile but only keep some fields
  - **`-select NAME,FIPS`**: keep only the NAME and FIPS fields

```
ogr2ogr -f "ESRI Shapefile" -t_srs
  "EPSG:4269" -select NAME,FIPS
  cache_towns_geo.shp cache_towns.shp
```

- Extract a subset of features from a shapefile
  - **`-where "cover='grass'"`** : keep only those features where the cover attribute is “grass”

```
ogr2ogr -f "ESRI Shapefile" -where
  "cover='grass'" grass.shp sites.shp
```

- Extract a subset of features within a certain extent
  - **`-spat 46000 4590000 490000 4600000`**: keep only those features that fall within the rectangle defined by **`xmin ymin xmax ymax`**

```
ogr2ogr -f "ESRI Shapefile" -spat 46000
  4590000 490000 4600000 somesites.shp
  sites.shp
```

- Convert a shapefile to KML
  - If there is a field called “name” in the shapefile, then it will be used to name the features in the KML, otherwise they will not be named
  - If there is a field called “description” in the shapefile, then it will be used as the feature description in KML, otherwise the description will be a table of all attribute values

```
ogr2ogr -f "KML" counties.kml
  ut_counties.shp
```

- Override name and description fields for KML output
  - -dsco NameField=id: uses the “id” field as the KML feature names
  - -dsco DescriptionField=cover: uses the “cover” field as the KML feature descriptions

```
ogr2ogr -f "KML" -dsco NameField=id -dsco
  DescriptionField=cover sites2.kml
  sites.shp
```

gdalinfo

- Prints info about a raster data set

```
Z:\data>gdalinfo
Usage: gdalinfo [--help-general] [-mm] [-stats] [-hist] [-nogcp] [-nomd]
                [-noct] [-checksum] [-mdd domain]* datasetname
```

- -nomd is my favorite flag because it suppresses the raster attribute table

```
Z:\data>gdalinfo aster.img
Driver: HFA/Erdas Imagine Images (.img)
Files: aster.img
       aster.rrd
Size is 5665, 5033
Coordinate System is:
PROJCS["UTM Zone 12, Northern Hemisphere",
    GEOGCS["WGS_1984",
        DATUM["WGS_1984",
            SPHEROID["WGS 84",6378137,298.2572235630016],
            TOWGS84[0,0,0,0,0,0,0]],
        PRIMEM["Greenwich",0],
        UNIT["degree",0.0174532925199433],
        AUTHORITY["EPSG","4326"]],
    PROJECTION["Transverse_Mercator"],
    PARAMETER["latitude_of_origin",0],
    PARAMETER["central_meridian",-111],
    PARAMETER["scale_factor",0.9996],
    PARAMETER["false_easting",500000],
    PARAMETER["false_northing",0],
    UNIT["Meter",1],
    AUTHORITY["EPSG","32612"]]
Origin = (419976.500000000000000,4662422.500000000000000)
Pixel Size = (15.000000000000000,-15.000000000000000)
Corner Coordinates:
Upper Left  (  419976.500, 4662422.500) (111d58'4.52"W, 42d 6'35.34"N)
Lower Left  (  419976.500, 4586927.500) (111d57'27.92"W, 41d25'47.74"N)
Upper Right (  504951.500, 4662422.500) (110d56'24.38"W, 42d 6'49.98"N)
Lower Right (  504951.500, 4586927.500) (110d56'26.64"W, 41d26'2.04"N)
Center      (  462464.000, 4624675.000) (111d27'5.89"W, 41d46'22.91"N)
```

```
Band 1 Block=64x64 Type=Byte, ColorInterp=Undefined
  Description = ASTER_Band1
  Min=1.000 Max=255.000
  Minimum=1.000, Maximum=255.000, Mean=76.339, StdDev=16.966
  Overviews: 1417x1259, 709x630, 355x315, 178x158, 89x79, 45x40
  Metadata:
    STATISTICS_MINIMUM=1
    STATISTICS_MAXIMUM=255
    STATISTICS_MEAN=76.338913470953
    STATISTICS_MEDIAN=77
    STATISTICS_MODE=81
    STATISTICS_STDDEV=16.965647083721
    LAYER_TYPE=athematic

    STATISTICS_HISTOBINVALUES=236|230|201|225|207|227|241|215|201|225|220|196|240|223|2
    02|204|222|231|199|218|209|224|21
<snip>
<GDALRasterAttributeTable>
  <FieldDefn index="0">
    <Name>Histogram</Name>
    <Type>1</Type>
    <Usage>0</Usage>
  </FieldDefn>
  <Row index="0">
    <F>236</F>
  </Row>
  <Row index="1">
    <F>230</F>
  </Row>
...
```

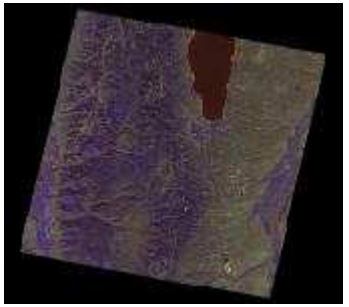
```
Z:\data>gdalinfo -nomd aster.img
Driver: HFA/Erdas Imagine Images (.img)
Files: aster.img
       aster.rrd
Size is 5665, 5033
Coordinate System is:
PROJCS["UTM Zone 12, Northern Hemisphere",
    GEOGCS["WGS_1984",
        DATUM["WGS_1984",
            SPHEROID["WGS 84",6378137,298.2572235630016],
            TOWGS84[0,0,0,0,0,0,0]],
        PRIMEM["Greenwich",0],
        UNIT["degree",0.0174532925199433],
        AUTHORITY["EPSG","4326"]],
    PROJECTION["Transverse_Mercator"],
    PARAMETER["latitude_of_origin",0],
    PARAMETER["central_meridian",-111],
    PARAMETER["scale_factor",0.9996],
    PARAMETER["false_easting",500000],
    PARAMETER["false_northing",0],
    UNIT["Meter",1],
    AUTHORITY["EPSG","32612"]]
Origin = (419976.500000000000000,4662422.500000000000000)
Pixel Size = (15.000000000000000,-15.000000000000000)
Corner Coordinates:
Upper Left  (  419976.500, 4662422.500) (111d58'4.52"W, 42d 6'35.34"N)
Lower Left  (  419976.500, 4586927.500) (111d57'27.92"W, 41d25'47.74"N)
Upper Right (  504951.500, 4662422.500) (110d56'24.38"W, 42d 6'49.98"N)
Lower Right (  504951.500, 4586927.500) (110d56'26.64"W, 41d26'2.04"N)
Center      (  462464.000, 4624675.000) (111d27'5.89"W, 41d46'22.91"N)
Band 1 Block=64x64 Type=Byte, ColorInterp=Undefined
  Description = ASTER_Band1
  Min=1.000 Max=255.000
  Minimum=1.000, Maximum=255.000, Mean=76.339, StdDev=16.966
  Overviews: 1417x1259, 709x630, 355x315, 178x158, 89x79, 45x40
Band 2 Block=64x64 Type=Byte, ColorInterp=Undefined
  Description = ASTER_Band2
  Min=1.000 Max=255.000
  Minimum=1.000, Maximum=255.000, Mean=62.817, StdDev=21.522
  Overviews: 1417x1259, 709x630, 355x315, 178x158, 89x79, 45x40
Band 3 Block=64x64 Type=Byte, ColorInterp=Undefined
  Description = ASTER_Band3N
  Min=1.000 Max=218.000
  Minimum=1.000, Maximum=218.000, Mean=68.693, StdDev=17.452
  Overviews: 1417x1259, 709x630, 355x315, 178x158, 89x79, 45x40
```

gdal_translate

- Convert between raster formats

```
Z:\data>gdal_translate
Usage: gdal_translate [--help-general]
       [-ot {Byte/Int16/UInt16/UInt32/Int32/Float32/Float64/
             CInt16/CInt32/CFloat32/CFloat64}] [-strict]
       [-of format] [-b band] [-expand {rgb|rgba}]
       [-outsize xsize[%] ysize[%]]
       [-scale [src_min src_max [dst_min dst_max]]]
       [-srcwin xoff yoff xsize ysize] [-projwin ulx uly lrx lry]
       [-a_srs srs_def] [-a_ullr ulx uly lrx lry] [-a_nodata value]
       [-gcp pixel line easting northing [elevation]]*
       [-mo "META-TAG=VALUE"]* [-quiet] [-sds]
       [-co "NAME=VALUE"]*
       src_dataset dst_dataset
```

- Convert to full-sized jpg (5665x5033, 2MB)
  - **`-of jpeg`**: output will be a jpeg
  - **`aster.img`**: input filename
  - **`aster1.jpg`**: output filename



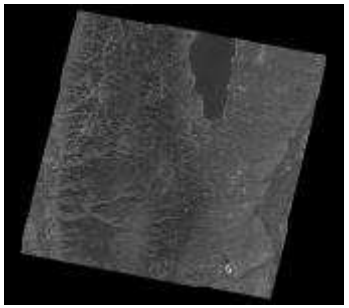
```
gdal_translate -of jpeg aster.img aster1.jpg
```

- Make a much smaller jpg (566x503, 35KB)
  - **`-outsize 10% 10%`**: output will be 10% the size of input

```
gdal_translate -of jpeg -outsize 10% 10%
  aster.img aster2.jpg
```

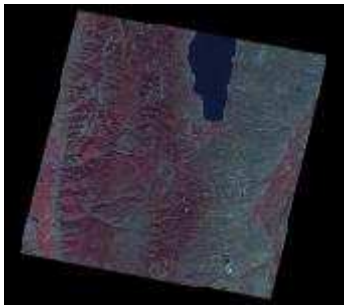
- Only use the first band in the image
  - **`-b 1`**: use band 1

```
gdal_translate -of jpeg -b 1
  aster.img aster3.jpg
```



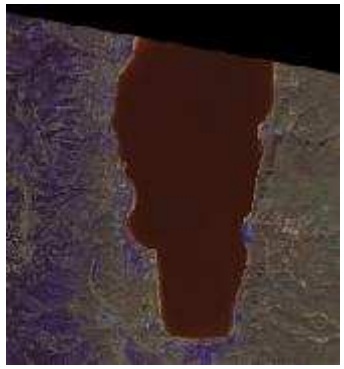
- Re-order the bands
  - **`-b 3 -b 2 -b 1`**: Put the bands in 3,2,1 order

```
gdal_translate -of jpeg
  -b 3 -b 2 -b 1 aster.img
  aster4.jpg
```



- Subset the image
  - **`-projwin 458000 4658000 483000 4631000`**: clip to the box defined by **`ulx uly lrx lry`**

```
gdal_translate -of jpeg -projwin 458000
  4658000 483000 4631000 aster.img
  aster5.jpg
```



gdaladdo

- Build overviews (pyramids) for an image

```
C:\>gdaladdo
Usage: gdaladdo [-r {nearest,average,gauss,average_mp,average_magphase,mode}]
                [-ro] [--help-general] filename levels

  -r : choice of resampling method
  -ro : open the dataset in read-only mode, in order to generate
        external overview (for GeoTIFF datasets especially)

Usefull configuration variables :
  --config USE_RRD YES : Use Erdas Imagine format (.aux) as overview format.
Below, only for external overviews in GeoTIFF format:
  --config COMPRESS_OVERVIEW {JPEG,LZW,PACKBITS,DEFLATE} : TIFF compression method
  --config PHOTOMETRIC_OVERVIEW {RGB,YCBCR,...} : TIFF photometric interpretation
  --config INTERLEAVE_OVERVIEW {PIXEL|BAND} : TIFF interleaving method

Examples:
 % gdaladdo -r average abc.tif 2 4 8 16
 % gdaladdo --config COMPRESS_OVERVIEW JPEG --config PHOTOMETRIC_OVERVIEW YCBCR
             --config INTERLEAVE_OVERVIEW PIXEL -ro abc.tif 2 4 8 16
```

- Build overviews (pyramids) for an image
- To build Imagine-style pyramids:
  - First create the image if needed:

```
gdal_translate -of hfa -projwin 458000 4658000
  483000 4631000 aster.img astersub.img
```

  - Then build the pyramids:
  - **`--config HFA_USE_RRD YES`**: use Imagine-style pyramids
  - **`3 9 27 81`**: the levels to build

```
gdaladdo --config HFA_USE_RRD YES
  astersub.img 3 9 27 81
```

gdalwarp

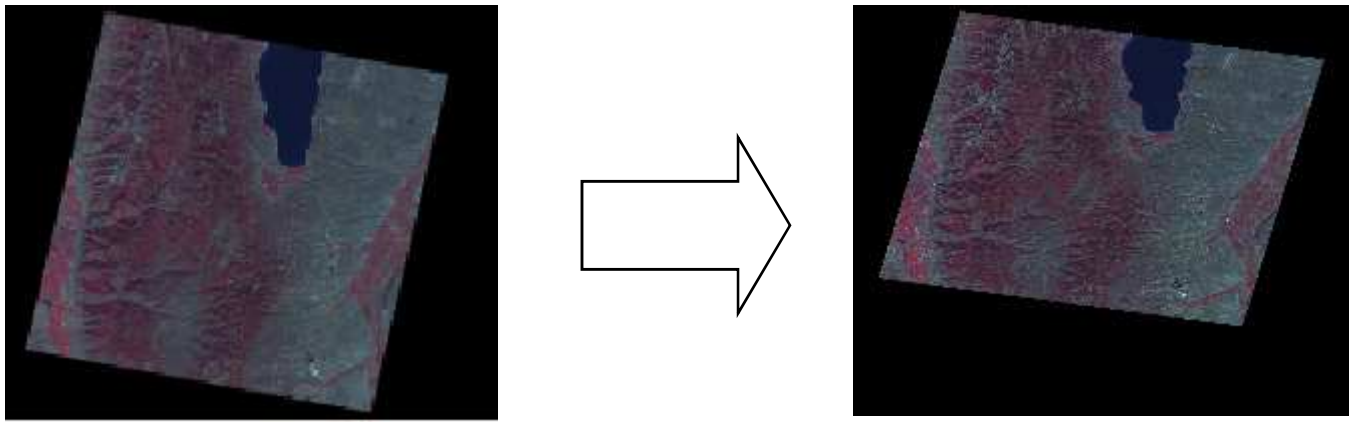
- Reproject images

```
C:\>gdalwarp
Usage: gdalwarp [--help-general] [--formats]
    [-s_srs srs_def] [-t_srs srs_def] [-to "NAME=VALUE"]
    [-order n] [-tps] [-rpc] [-geoloc] [-et err_threshold]
    [-te xmin ymin xmax ymax] [-tr xres yres] [-ts width height]
    [-wo "NAME=VALUE"] [-ot Byte/Int16/...] [-wt Byte/Int16]
    [-srcnodata "value [value...]"] [-dstnodata "value [value...]"] -
   dstalpha
    [-r resampling_method] [-wm memory_in_mb] [-multi] [-q]
    [-cutline datasource] [-cl layer] [-cwhere expression]
    [-csql statement] [-cblend dist_in_pixels]
    [-of format] [-co "NAME=VALUE"]*
    srcfile* dstfile

Available resampling methods:
    near (default), bilinear, cubic, cubicspline, lanczos.
```

- Reproject images
  - **`-of hfa`**: output will be an Imagine file
  - **`-t_srs EPSG:4326`**: output will be geo WGS84
  - **`-r bilinear`**: use bilinear interpolation

```
gdalwarp -of hfa -t_srs EPSG:4326 -r
  bilinear aster.img aster_geo.img
```



gdal_merge.py

- Python script to mosaic images

```
C:\>gdal_merge
No input files selected.
Usage: gdal_merge.py [-o out_filename] [-of out_format] [-co NAME=VALUE]*
                     [-ps pixelsize_x pixelsize_y] [-separate] [-v] [-pct]
                     [-ul_lr ulx uly lrx lry] [-n nodata_value] [-init value]
                     [-ot datatype] [-createonly] input_files
                     [--help-general]
```

- Only works with some output formats (not jpeg or png, for example)

- Mosaic two images
  - **`-o mosaic1.tif`**: output filename
  - **`-of gtiff`**: output will be GeoTIFF
  - **`-co tfw=yes`**: create a worldfile
  - **`q0519ne.jpg q0520nw.jpg`**: input files

```
gdal_merge -o mosaic1.tif -of gtiff -co
  tfw=yes q0519ne.jpg q0520nw.jpg
```

- Mosaic and subset at the same time
  - **`-ul_lr 424600 4622700 430000`**: clip out the box defined by ulx uly lrx lry

```
gdal_merge -o mosaic2.tif -of gtiff -co
  tfw=yes -ul_lr 424600 4622700 430000
  4620000 q0519ne.jpg q0520nw.jpg
```

- Make a reduced resolution mosaic
  - **`-ps 100 100`**: output pixels will be 100 units wide by 100 units tall

```
gdal_merge -o mosaic3.tif -of gtiff -co
  tfw=yes -ps 100 100 q0519ne.jpg
  q0520nw.jpg
```

- Mosaic lots of files at once
  - **`--optfile files.txt`**: pass in a text file with filenames to mosaic

```
gdal_merge -o mosaic4.tif -of gtiff -co
  tfw=yes -ps 100 100 --optfile files.txt
```

where files.txt has filenames, one per line:

```
q0519ne.jpg
q0520nw.jpg
...
```

Calling other programs from Python

- **os.system(<command_string>)**

```
import os
os.system('gdaladdo --config HFA_USE_RRD YES
  q0520nw.img 3 9 27 81')
```

- Batching it:

```
import glob, os
for fn in glob.glob('d:/data/*.img'):
  os.system('gdaladdo --config HFA_USE_RRD
  YES ' + fn + ' 3 9 27 81')
```

- Open ArcMap and wait for it to close (use / or \\ for directory separators)

```
os.system('C:/Progra~1/ArcGIS/Bin/ArcMap.exe')
```

```
os.spawnl(os.P_WAIT, 'C:/Program
  Files/ArcGIS/Bin/ArcMap.exe')
```

- Open ArcMap and let Python continue on its way

```
os.spawnl(os.P_NOWAIT, 'C:/Program
  Files/ArcGIS/Bin/ArcMap.exe')
```

- Open all .py files in a directory with Crimson Editor (use Windows notation \ for passed arguments)

```
os.spawnl(os.P_NOWAIT, 'C:/Program
  Files/Crimson Editor/cedt.exe',
  'cedt.exe',
  'd:\data\classes\python\os8\*.py‘)
```

Opening files

- Windows only
- The file extension must have an associated application
- Python has no idea what you do with the file once it's open
- Open a Word document

```
os.startfile('test.doc')
```

Using GDAL & OGR with ArcGIS

- Need to install GDAL, Numeric, etc for the ArcGIS version of Python (see the installation instructions on the course website)
- Can import arcgisscripting , gdal, ogr, Numeric, etc all in the same python script
- Can run script from PythonWin, Crimson Editor (if you configure it to run with the ArcGIS version of Python), the command prompt, or a toolbox

- If running from a toolbox, set the input as a Feature Class (vector) or Raster Dataset (raster). If you use Feature Layer or Raster Layer, then the user can select something loaded into ArcMap, but ArcMap will not pass the full filename to the script and it will probably die

- Seems to work great with ArcMap 9.3, but…
- I ran into these problems last year with ArcMap 9.2:
  - Can run from a toolbox, although it is slower than running it other ways and you have to import arcgisscripting even if not using it
  - If you make the output raster a RasterDataset then ArcGIS makes it temporary and deletes it automatically!

- If you make the output raster a RasterLayer then it isn't added to ArcMap automatically
- I got the script to run in a model when it was the ONLY thing in the model
- I tried using the script's output as an input to another tool in a model, and ArcGIS died a horrible death (without even trying to run the script! – when the user chooses the input and output files)

Assignment 7

- Convert a shapefile to KML
  - Use “STATE_NAME” as the name field for the KML features
  - Include all of the attributes in the description
  - Turn in the command you used to do it and a screenshot of your KML being displayed in Google Earth

The end

- Evaluations...