

LANDSCAPE MODELS WITH PYTHON, ARCPY, PANDAS, GEOPACKAGE AND SPATIALITE

ESRI User Conference 2016



INTRODUCTION

- Currently Planning Analyst Helena-Lewis and Clark National Forest
- GIS Program Manager Bighorn National Forest
- Systems Analyst Georgia-Pacific and Olympic Resource Management

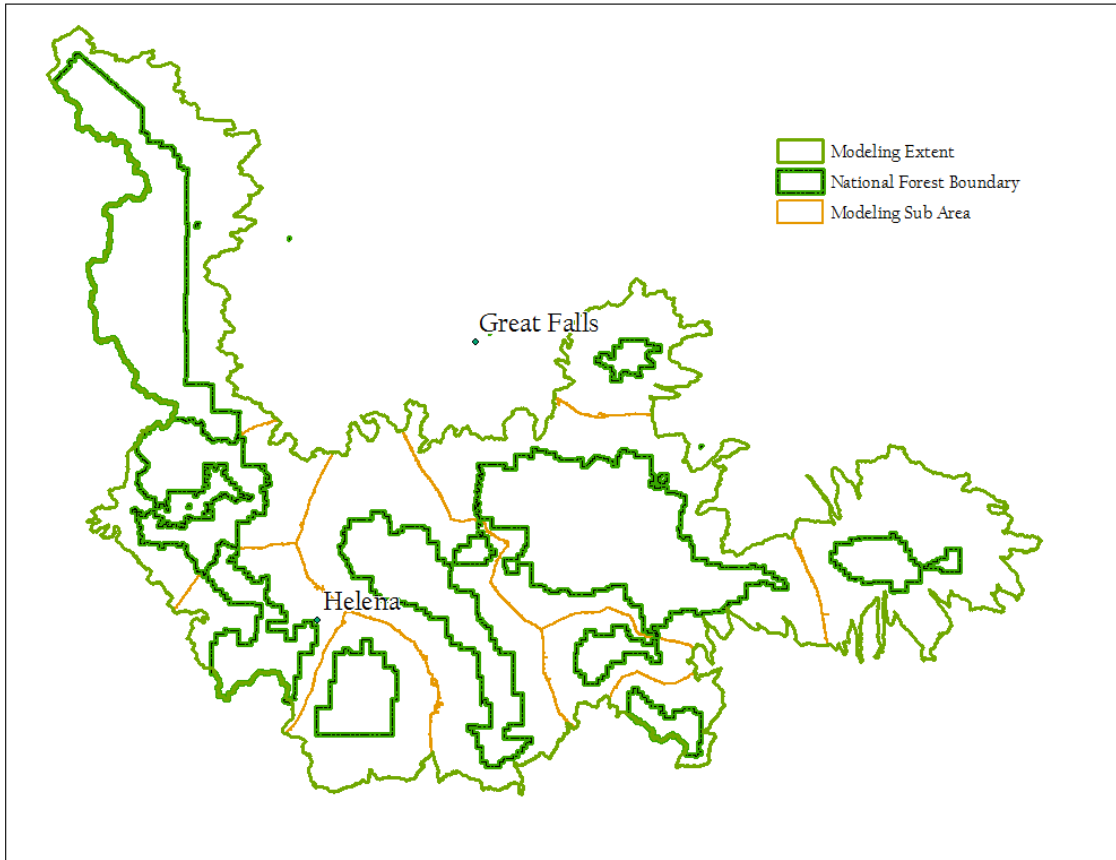


TOPICS THAT WILL BE DISCUSSED

- Landscape Analysis
- Tools for Performing Landscape Analysis
- Data Formats for Landscape Analysis
- Examples of Using the Tools
- Conclusion



HELENA-LEWIS AND CLARK NATIONAL FOREST MODELING AREA

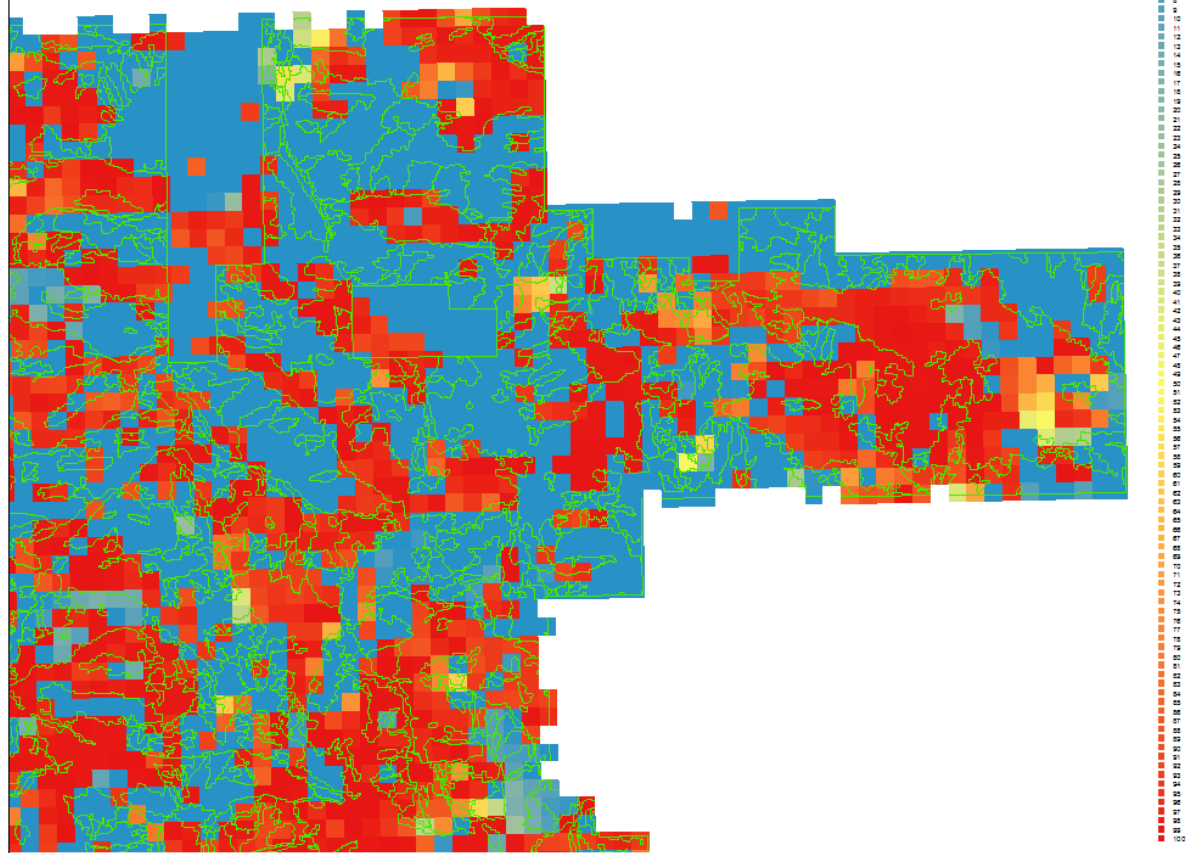


MODELING NATURAL RANGE OF VARIATION

- Goal is to create an estimate of the variation in vegetation and disturbance processes occurring over the last 1,000 years.
- 30 runs are done.
- This generates a lot of data ~ 234 Gigabytes



Presence of Douglas-fir



SPECIES PROBABILITY



SIMPPLLE

- SIMulating Patterns and Processes at Landscape LEvel
- A spatially explicit landscape level vegetation simulation model that include stochastic processes such as fire and insect and disease outbreaks and regeneration.
- Data architecture is circa 1995



MODELING WORKFLOW

THE IDEAL VIEW OF TIME DISTRIBUTION

Multiple
Input
Data
Sources

SIMPPLLE

Spreadsheets,
Graphs and Maps



MODELING WORKFLOW

THE ACTUAL TIME DISTRIBUTION

Multiple Input
Data Sources

SIMPPLLE

Spreadsheets, Graphs
and Maps



PYTHON – THE BIG TOOLBOX

- The glue holding it all together
- Many GIS tools that are available for Python
- Important to have a solid Python distribution



PYTHON – ARCGIS DISTRIBUTION

- Upgrades to ArcGIS create a new installation that does not preserve the previous package configuration
- Packages can be old
- Does not have full suite of scientific packages



PYTHON - DISTRIBUTION

- Several scientific distributions
 - Anaconda
 - Python (X,Y)
 - WinPython
- Many packages already installed
- Package management to handle version dependencies
- Come with several IDE's including IPython and Jupyter Notebooks



PYTHON TOOLS USED

- Arcpy
- Pandas
- Spatialite
- Geopandas



ARCPY

- Integrates with ArcGIS
- Numerous tools available

- Lacking in large scale data management such as provided by SQL database
- Not easily accessible with non ESRI tools



ADDING ARCPY TO PYTHON DISTRIBUTION

- 32 Bit Distribution
- Copy desktop10.2.pth into ...\\Lib\\site-packages

- 64 Bit distribution
- Copy DTBGGP64.pth to ...\\Lib\\site-packages



SPATIALITE

- Uses Sqlite database format and open source libraries
- Available since 10.2 as a native ESRI format
- Accessible from Python and numerous third party tools
- It is a true database



USING SPATIALITE IN ARCPY

- Import Spatialite Extension (DLL) into Python
- Can use ESRI driver – documentation is incomplete
- Just one binary package includes all libraries
- Can use Spatialite driver – better documentation and more functions



HOW TO LOAD SPATIALITE EXTENSION

- `import sqlite3`
- `db_file = r'F:/Path2Database/GIS_Database.sqlite'`
- `db_conn = sqlite3.connect(db_file)`
- `db_conn.enable_load_extension(True)`
- `db_conn.execute("select load_extension('mod_spatialite')")`



GEOPACKAGE

- Use SQLite database format
- Lightweight data transfer
- Open GIS specification
- Supported by many tools – Check website



SPATIALITE VS GEOPACKAGE

- Geopackage is a new OGC standard
- Spatialite has been around for many years
- Spatialite natively support Geopackage data with numerous functions.
- Queries using Spatial indexing on Geopackages use older Rtree syntax instead of new Spatialite SpatialIndex syntax
- There are some advanced functions such as ElementaryGeometry and KNN that are not supported for Geopackage data



CREATING ESRI COMPATIBLE DATA IN SPATIALITE/GEOPACKAGE

- Used when creating a spatial table from another spatial table by applying spatial operator (buffer, intersect, etc)
- Spatialite has a RecoverGeometryColumn function to get full database functionality
- ArcGIS has a higher standard for defining a table with spatial data



CREATING ESRI COMPATIBLE DATA IN SPATIALITE/GEOPACKAGE

- Database file name must end with `.SQLITE` or `.GPKG`
- Requires a defined primary key field
- SRID/ projection required to be stored with each geometry value
- Requires that the field storing the geometry have a defined data type.



CREATING ESRI COMPATIBLE DATA IN SPATIALITE/GEOPACKAGE

- create table my_points (pnt_id integer primary key autoincrement not null,name text')
- select
gpkgAddGeometryColumn('my_points','the_geom','POINT',0,0,26912)')
- select
gpkgAddGeometryTriggers('my_points','the_geom')
- insert into my_points(name,the_geom) values ('use
gpkgMakePoint',gpkgMakePoint(x_coor,y_coor,26912))
- select pkgAddSpatialIndex('my_points','the_geom')



PANDAS

- Python library for data manipulation and data analysis
- Comparable to R but easier to learn
- Ties into Python scientific and data science stack



GEOPANDAS

- Power of Pandas applied to Spatial data
- Reads shapefiles, file geodatabases and Spatialite



COLLAPSE POLYGONS EXAMPLE

- Starting layer is the union of 5 other layers
- Create a layer that only contains polygons that in aggregate total more than 30 acres



COLLAPSE POLYGONS EXAMPLE

```
import arcpy
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
%matplotlib inline
```



COLLAPSE POLYGONS EXAMPLE

- Import libraries that will be used

```
import arcpy
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
%matplotlib inline
```



COLLAPSE POLYGONS EXAMPLE

- Setup some variables

```
arcpy.env.overwriteOutput = True  
min_aaname_size = 30  
hard_code_field_names=('OBJECTID', 'aaname', 'SHAPE@AREA')
```



COLLAPSE POLYGONS EXAMPLE

- Read in the data

```
gdb_file = r'C:\workspace\ESRI_UC\Spectrum_Base.gdb'  
original_spectrum=gdb_file + os.sep + r'AllLayers_wSYL'  
spec_df=pd.DataFrame(arcpy.da.FeatureClassToNumPyArray(original_spectrum,hard_code_field_names))
```

- `spec_df=pd.DataFrame(arcpy.da.FeatureClassToNumPyArray(original_spectrum,hard_code_field_names))`



COLLAPSE POLYGONS EXAMPLE

- Group the data and select out data of interest

```
aaname_group=spec_df.groupby(['aaname'])['SHAPE@AREA'].sum()
aaname_group.sort()
aaname_group=pd.DataFrame(aaname_group).reset_index()
aaname_group_good = aaname_group[aaname_group['aaname'].str.contains("X")==False]
aaname_group_bad = aaname_group[aaname_group['aaname'].str.contains("X")]

del_aaname_df = aaname_group_good[aaname_group_good['SHAPE@AREA']<min_aaname_size*4046.856]
aaname_to_eliminate = list(del_aaname_df.aaname.values)
```



COLLAPSE POLYGONS EXAMPLE

- Get rid of real sliver polygons

```
spectrum_layer = 'spectrum_layer' # dummy variable
arcpy.management.MakeFeatureLayer(original_spectrum, spectrum_layer)
arcpy.management.SelectLayerByAttribute(spectrum_layer, 'CLEAR_SELECTION')
arcpy.management.SelectLayerByAttribute(spectrum_layer, 'NEW_SELECTION', "shape_area<1")
spectrum_sliver = gdb_file + os.sep + r'AllLayersUnioned_LessSliver'
arcpy.management.Eliminate(spectrum_layer, spectrum_sliver, 'LENGTH')
```



COLLAPSE POLYGONS EXAMPLE

- Create selection of groups to eliminate, then do the Eliminate command

```
spectrum_deslivered = 'spectrum_deslivered'
arcpy.management.MakeFeatureLayer(spectrum_sliver,spectrum_deslivered)
arcpy.management.SelectLayerByAttribute(spectrum_deslivered,'CLEAR_SELECTION')

for del_au in aaname_to_eliminate:
    where_clause = "aaname = '%s'%del_au
#     arcpy.management.SelectLayerByAttribute(spectrum_layer,'ADD_TO_SELECTION',where_clause)
    arcpy.management.SelectLayerByAttribute(spectrum_deslivered,'ADD_TO_SELECTION',where_clause)
    desc=arcpy.Describe('spectrum_deslivered')
    au_new = gdb_file + os.sep + r'AllLayersUnioned_Gt20Acres_deslivered'
#arcpy.management.Eliminate(spectrum_deslivered,au_new,'AREA')
arcpy.management.Eliminate(spectrum_deslivered,au_new,'LENGTH')
```



COLLAPSE POLYGONS EXAMPLE

- Check what happened.

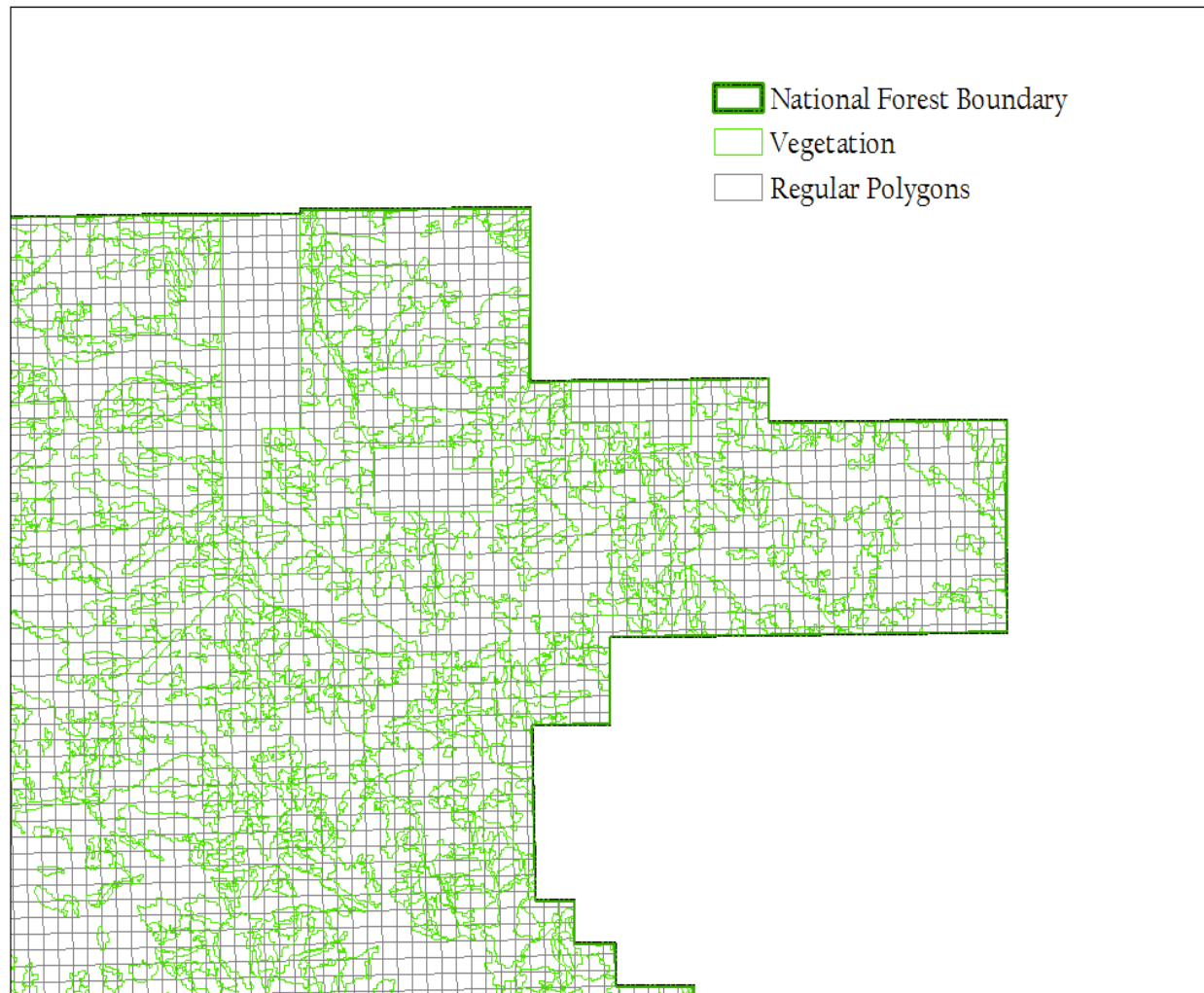
```
au_slim_down_df=pd.DataFrame(arcpy.da.FeatureClassToNumPyArray(au_new,hard_code_field_names))
au_group=pd.DataFrame(au_slim_down_df.groupby(['aaname'])['SHAPE@AREA'].sum())
au_group.hist()
au_group.sort(columns=['SHAPE@AREA'])[:10].plot()
```

- Repeat the selection and eliminate if needed.

-



SIMPLE DATA PREP



CONCLUSION

- There are number of tools available for spatial data processing
- A modeling process can be created by using a combination of tools.



REFERENCES

- **SIMPPLLE** –
<http://www.fs.fed.us/rm/missoula/4151/SIMPPLLE/>
- **Spatialite** – <http://www.gaia-gis.it/gaia-sins/>
- **Geopackage** – <http://www.geopackage.org/>



REFERENCES

- **Pandas** - <http://pandas.pydata.org/>
- **GeoPandas** - <http://geopandas.org/> & <https://github.com/geopandas/geopandas>



CONTACT

davidanderson@fs.fed.us

406-495-3744

