Solar Site Selection Using Weighted Linear Combination in Jefferson County, Colorado

GEOG 4303 Final Project Report - May 7th, 2019

Trevor Stanley, Xingying Huang, and Avra Saslow

Section 1: Introduction and Motivation

While the majority of electricity generated in Colorado is sourced from fossil fuels, the state's progressive agenda to increase the renewable fuel portfolio standard percentage over the last few years necessitates increased renewable energy sources. Furthermore, the cost of both wind and solar power per kWh, when combined with battery storage, is now cheaper than electricity generated by fossil fuels (GreenBiz, 2018). The primary mechanism by which a municipality or utility develops and acquires solar projects is by issuing a Request For Proposals (RFPs) for prospective developers to bid into. Typically the window of time to bid into these RFPs is between two to six weeks. Hence, a developer must quickly identify and lock in a large number of suitable sites in order to compete for the RFP. This poses a large challenge, especially for smaller developers, when the service territory of the RFP issuer is large (e.g. it covers many counties or even states). Developers must parse through large datasets and spend large amounts of time trying to manually identify suitable locations via a tool like Google Earth, which lacks critical data like the parcel number and owner information.

This challenge of finding a large number of sites in a small amount of time that are suitable to develop a solar system on begets the need for a tool to automate this process. Such a tool must be capable of providing different levels of granularity on parameters that constrain site selection. Our team set out to create a model that is capable of doing just this. With over 463 solar companies in Colorado alone, there are a large number of potential customers we can market such a tool to, with Colorado as our beachhead market (SEIA, 2018).

Section 2: Objectives/Deliverables

The main objective and deliverable of this project is to develop python model within ArcPy that can be hosted on and integrated with a locally hosted website that takes input of county parcel data of any standard file type, such as KMLs, Shape, or Raster files. The backend code in ArcPy will then automate the process of standardizing the data. Specifically, this will mean standardizing naming conventions, projections, raster cell sizes, and format such that the data is compatible with the various modules and functions of the model. The primary goal is to automate prospecting for solar arrays larger than 6 MW, translating to over 30 acres of land. These data sets will be uploaded by the user to a source data folder.

The adjustable criteria in this model are: parcel size (in acres), the percentage of the focal window for aspect, slope, and vegetation. These need to be altered directly in the script by the user.
Non-negotiable criteria included in the model are distance from electric infrastructure, a parcel size of at least 30 acres, parcels outside of a floodplain, and certain municipal zoning classes. The specific approach we employed will be detailed in the methods section. Finally, our team wanted to deliver a model that was clean and easy to follow along for someone familiar with ArcGIS and Python, but who is not an expert.

Section 3: Methods

3.1 Background & Overview

3.1.1 Study Area and Data

The original five county datasets selected for this study were within Colorado: Jefferson county, Montezuma County, Weld County, Arapahoe County, and Garfield County. Colorado, generally, represents an ideal study area, as it has an average of 300 days of sunshine, over 463 solar companies, and a wealth of geographic data. Geographic data was sourced from county government websites as well as the Colorado Atlas dataset supplied by the Geography Department at the University of Colorado Boulder. The data selected from counties included parcel and zoning shapefiles. The data selected from the Colorado Atlas included the NLCD raster and the Colorado DEM, or elevation, raster. The five counties initially selected were filtered to just Jefferson County. Jefferson County was selected because it was the most representative county of the five in terms of the mix of rural and urban areas, as well as different topographies. The relevant attribute fields pertaining to this study can be found in [Appendix A].

3.2 Functions

3.2.1 Projections and Rasters

Upon collecting the relevant data, all projections were standardized to WGS 84 UTM 13N, as this was the standard projection used for the Colorado DEM and NLCD datasets. Once the projections were standardized, all rasters were resampled to 120 meter cell size. This larger cell size enabled faster data processing and ultimately better suited the parcel sizes we wanted to asses (e.g. parcels larger than 30 acres). Since this analysis focuses on parcels that are larger than 30 acres, a larger cell size is ideal as there is likely minimal extreme variation of things like slope or vegetation within a parcel, especially after smoothing via the focal mean window function (described in more detail below).

3.2.2 Area

Parcel size or area was selected for using a simple update cursor. The cursor searches the shape area and then a query is performed to find those rows within the shape area column that have an area less than 30 acres, or 1307000 square feet. If this condition is met, the row is deleted from the overall shape file. See [Appendix B] for the code.

3.2.3 Municipal Zoning

Utility scale solar arrays can only be developed in certain zones; namely agricultural (A) districts and planned development (P-D) parcels. These municipal zones were queried for in our model, but in doing so, we accepted a certain degree of uncertainty introduced in the model. Some of the P-D parcels weren't fully selected, as they shared boundary lines with other parcels took majority precedence of the polygon shape. Therefore, in some cases, the topology didn't fully render, and only lines of P-D were selected - not the entire polygon. Thus, there is some uncertainty introduced into the model. See [Appendix D] for examples.

3.3 Focal Window and Suitability Rasters

3.3.1 Focal Mean Window

This function smooths the array such that the cells within the window are reassigned a percentage based on the ratio of 1's and 0's within the focal window. The window accounts for edge cases, and has and adjustable cell size. See [Appendix E] for the code.

3.3.2 Suitability Rasters - NLCD Raster Land Coverage Analyzation

The main goal of this section is to eliminate unsuitable sites for large-scale PV solar sites selection such as developed high intensity commercial zone, or forest area dominated by trees generally greater than 5 meters tall. The National Land Cover Database (NLCD) 2011 provides land cover classification for Jefferson County, CO to select sites that has minimal existing structures or development at the native 30-m spatial resolution of the Landsat Thematic Mapper (TM). While multiple approaches could be utilized to select only those parcels that don't have any existing developments like structures and roads, NLCD data provides a high level of accuracy and is easy to query. After clipping the NLCD raster to the Jefferson county layer that was produced from the previous functions, the raster is converted into a numpy array. A where clause is then performed on the arry to select for all the NLCD classes of interest (52 - shrub/Scrub; 71 - Grassland/Herbaceous; 81 - Pasture/Hay; 82 - Cultivated Crops). If the given cell, or array element, is one of these values, the cell/array element is reassigned a 1. If it isn't one of these values, it is reassigned a 0. This is then passed into a focal mean function which smooths the array such that the cells within the window are reassigned a percentage based on the ratio of 1's and 0's within the focal window. This new array is then queried and reassigned a 1 if a given cell/array element is above a certain specified threshold. See [Appendix F] for the code.

3.3.3 Suitability Rasters - Slope and Aspect

Creating suitability rasters relies on a similar methodology as weighted linear combination (WLC), whereby the cell values from multiple grids are multiplied by some weighting factor and are combined to form a single output grid [Appendix J]. Our team combines WLC with a pre-smoothing by way of a focal mean function with a moving window. Similarly to how the suitability raster for NLCD was performed, the Colorado DEM was clipped to the modified Jefferson County layer and turned into a numpy array. The ArcPy slope function was then used to determine slope and a slope between 0 and 8 degrees was selected for. See [Appendix G] for the code. In the same way as NLCD and Slope were selected for, the Colorado DEM was clipped and turned into a numpy array. The ArcPy aspect function was then used to assign an aspect value to each cell. A boolean grid was then created from this output with cells having an aspect of 90-270 being assigned an 1 and all other cells being assigned a 0. After being passed to the focal window function, another boolean grid is created based on the threshold percentage value. See [Appendix H] for the code.

3.4 Dynamic Floodplain

The creation of a floodplain function was the consequence of the Federal Emergency Management Agence (FEMA) not providing a floodplain data file for the counties of interest. This function therefore attempts to recreate a floodplain layer. Floods will spread more vastly in flat areas than in canyons. Therefore, the principle of the function was that it executed different (or dynamic) buffers based off of the elevation spatial autocorrelation of a river. A river surrounded by steep canyons would require less of a buffer than a river in the plains.

The original method for this implementation was calculating the least accumulative cost distance for each river cell based off of the Colorado digital elevation model through the Cost Distance tool. This created a raster that grouped cells in eight classes based on increase in elevation - 0 had the least change in elevation and 8 and the greatest change in elevation. From there, the intention was to find each pixel’s coordinates using Numpy and the Extent property, and create an ArcPy point based off of the coordinates. We could then apply the buffer for every point within a specific class such that there would be a larger buffer for classes of a lower value (0) and a smaller buffer for classes of a higher value (8). The final step would be dissolving all of the buffers together so that there would only be one layer remaining. This method would introduce uncertainty, as the buffers would overlap, so the idea was to work from class 8 down to class 0 so that the larger buffers would still take precedence and protect our parcels from a potential flood.

However, the Cost Distance function produced a raster that was more like an aspect raster than an output cost distance raster such that we weren't able to interpret the result at all (Appendix I). Other Arc Distance tools (i.e. path distance) were also unhelpful and incorrect in their outputs. After spending quite some time on the ever expanding complexities of the problem, the team decided to break up the dynamic floodplain based off of if where the rivers were located: in the plains or the mountains. Those segments in the plains received a larger buffer and those in the mountains received a smaller buffer.

3.5 Electric Infrastructure: Euclidean Distance vs. Buffer

A crucial component of the model is distance to electric infrastructure. Parcels within two miles of electric infrastructure (i.e. transmission lines and substations) were selected. Initially, this problem was approached by utilizing the Euclidean distance function - it describes each cell's relationship to a respective transmission line or substation based on a straight-line distance. However, this approach was logically inefficient, as we knew what distance we wanted: 2 miles. Finding each cell's distance to substations and power lines and then using a moving window to assign 1's to those within 2 miles and 0's to those outside 2 miles (using Numpy) was time costly for both the computer and the group members and their associative tasks in the project. Thus, the team decided to create a simple 2 mile buffer for each layer, that efficiently executed the needs to meet the criteria.

3.6 Data Visualization: Pandas and Plotly

One of the desired deliverables for the project was to create a tool in which developers could understand quickly and manipulate for their needs effectively. This was implemented through a platform called Dash, which is an open source framework built on Flask (a local server host), Plotly.js, and React that enables users to build web applications using pure Python.

A final raster of the selected sites were exported to a table and fed in through a Pandas dataframe such that the information could be easily plotted with Plotly. We attempted to load in shapefiles, but ran into problems in requiring a mapbox_key to utilize various functions. Thus, we decided to select specific columns from the dataframe - latitude, longitude, and plot number - and plot them with Plotly.

One challenge in doing so was that our data was in terms of easting and northing rather than latitude and longitude. Thus, we had to utilize a python library called utm which converts a UTM coordinate into a latitude and longitude tuple. It was particularly challenging integrating the coordinate conversion with the dataframe and calling from the dataframe to map out the points on the map. Although the points are plotted on the map, they don't have a text hover feature that shows their respective rank because that column (MAX) in the dataframe wasn't preserved in the conversion process. Further work needs to be done in preserving the MAX column, as well as importing the actual shape file so that the user can see the size of the parcel and not just the centroid. Appendix [L] shows the outputs from the model as displayed in the Plotly environment. The team was unable to get the coordinate mapping to work properly before submitting the project.

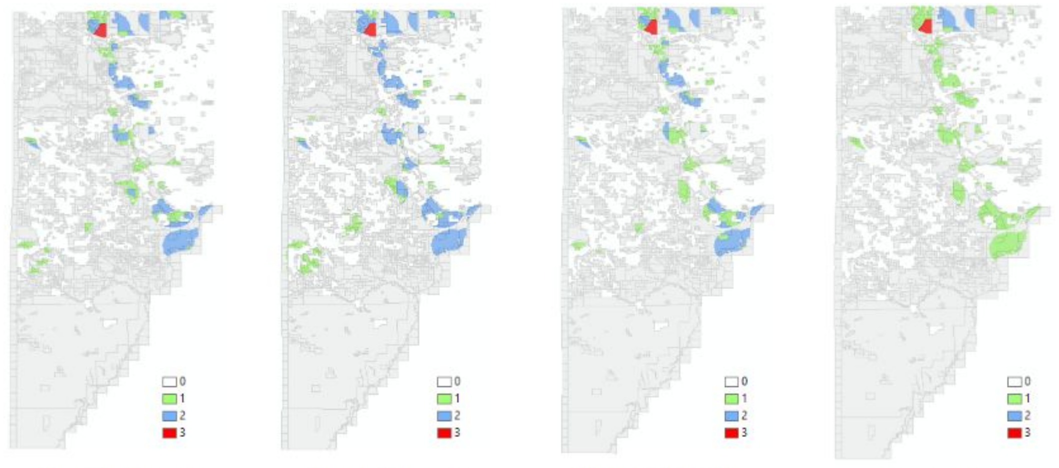
Section 4: Evaluation

Our team decided to rank selected sites via differential aspect ratios. Aspect was selected as it represented the largest range of possible values out of all of the other negotiable criteria. Slope, for instance, is already selected to be below 10%, and thus a 40% vs. 80% selection of the smoothed numpy array (produced from the focal mean function) will yield very small changes to the final output. Aspect, in contrast, has a large starting range. Hence, changing the selection of the smoother numpy array from say, 30% to 60%, will yield a much larger change in the final output. Three different aspect ranges were chosen such that the outputted parcels had an automated model ranking. These ranges are 30%, 60%, and 90%, which translated to the following rankings:

3 - an ideal site meeting all the specified parameters; aspect ratio 90%; passes subjective evaluation
2 - a good site; aspect ratio of 60%; passes subjective evaluation
1 - is a decent site; aspect ratio 30%; passes subjective evaluation

Finally, a ranking of 0 means that one or more of the critical criteria have not been met, and the site has been excluded.

Section 5: Results/Findings



50% NLCD, 50% slope, 50% aspect (control)

50% NLCD, 50% slope, 30% aspect

50% NLCD, 50% slope, 60% aspect

50% NLCD, 50% slope, 90% aspect

A final DEM is created and joined with relevant data from the original parcel layer. The grid cell values are included in this final DEM in an attribute field named MAX_12. These grid values reflect the ranking system/values discussed in the previous section. As the aspect threshold is changed, the number of sites meeting multiple criteria diminish. At a 90% aspect, few ideal sites are selected, with only two parcels meeting all strict criteria. Ultimately, when suitability thresholds are set at 50% (the control), the number of sites goes from over 28,000 to about 371.

Conclusion & Future Work

One major source of uncertainty in this model can be attributed to the method utilized for estimating the floodplain layer for Jefferson County. There is likely a large amount of error, with the floodplain being either much larger or much smaller in certain areas depending on actual elevation. Another source of error pertaining to the suitability grids is bias introduced from extreme changes in a given feature. Regions with drastic change in elevation or some other feature might bias a focal mean window up or down. Hence, the final boolean grid might mark areas that should be a 1 or 0 a 0 or 1, respectively.

Some of the selected suitable sites were within zoned parcels called P-D, or Planned Development. While these were important to keep in our model because there are many P-D parcels that are suitable for solar development, the act of including them also introduces uncertainty in the model. There are parcels of P-D that *aren't* suitable for solar development, or land could either be already developed or bought with the intention of development and it would still show up as a suitable site in our model. While the inclusion of P-D makes the confidence interval of our selection sites larger, it also makes it wider, which means it is less accurate and more general. Furthermore, outdated data contributed to the uncertainty in our model, as all of the datasets were at least eight years old and don't account for the numerous spatial and temporal changes in development and topography in Jefferson County.

Since bidding for a solar site is highly constrained by time and our tool intends to make the process easier, introducing uncertainty in the model is counterproductive to our goal. Therefore, future work could be allocated in making an uncertainty model that accompanies the tool and is dynamically built such that it shows the variety of uncertainty introduced in the model. It would simulate every scenario in which uncertainty exists and the variation of uncertainty - for example it would dynamically show how old datasets produce different parcel locations than new datasets, and how the inclusion of P-D changes the model. and simulates every scenario in which the inputs of uncertainty. Furthermore, additional work is needed on validating the results of this model. This could be done through going through all of the 371 outputs from the model and subjectively evaluating them. However, the ideal approach to validation would be through finding a dataset where the results have already been established/validated and running this original data through our model. We could then assess the differences in output and get an approximation for how much error is present in our model.

Finally, it is our goal that our model is extensive and at least can be run for all counties in Colorado. To attain that goal, we would need to do further work in generalizing parcel naming conventions across all counties such that they could easily be run through the model in the same manner.

References

Chaves, Andrea, and Terry Bahil. "ArcUser Online." Locating Sites for Photovoltaic Solar. Esri. 16 Apr. 2019

“Colorado Solar.” *SEIA*, Solar Energy Industries Association, 2018.

Rothberg, Daniel. “Here Comes the Sun: Solar plus Storage Energy Solutions Get Competitive.” *GreenBiz*, GreenBiz Group Inc., 25 June 2018,

Jefferson County Government. Maps and Data Download. 2019.

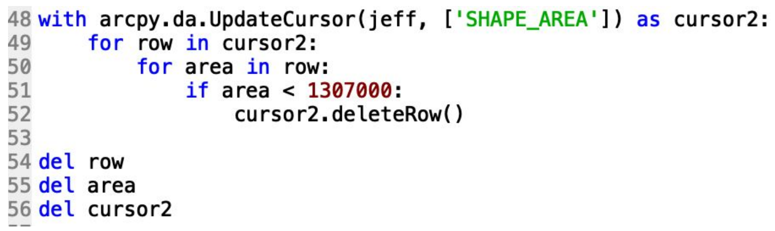
Uyan, Melvut. "GIS-based solar farms site selection using analytic hierarchy process (AHP) in Karapinar region, Konya/Turkey." Renewable and Sustainable Energy Reviews. 07 Aug. 2013. ScienceDirect. 16 Apr. 2019

Wanderer, Thomas, and Stefan Herle. “Creating a Spatial Multi-Criteria Decision Support System for Energy Related Integrated Environmental Impact Assessment.” *Environmental Impact Assessment Review*, ScienceDirect, 16 Sept. 2014.

Appendix

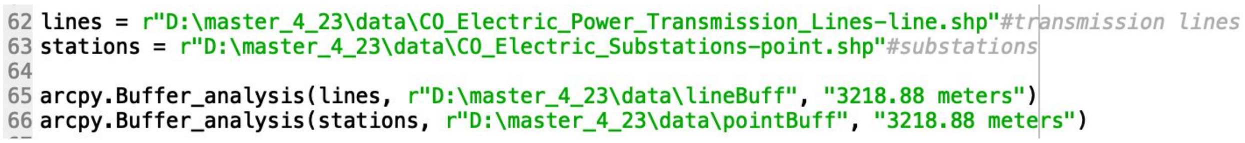
A. These correspond to the tax parcel ID, geographic coordinates, both the normal and property owner information, and the parcel area in square feet.
`["SPN","X_COORD","Y_COORD","OWNNAM","OWNNAM2","MAILSTRNBR","MAILSTRDIR","MAILSTRNAM","MAILSTRTYP","MAILSTRUNT","MAILCTYNAM","MAILSTENAM","MAILZIP5","PRPSTRNBR","PRPSTRDIR","PRPSTRNAM","PRPSTRTYP","PRPSTRUNT","PRPCTYNAM","PRPSTENAM","PRPZIP5","NHDNAM","SHAPE_AREA"]`

B. Remove parcels less than 30 acres (1307000 square feet) using a cursor



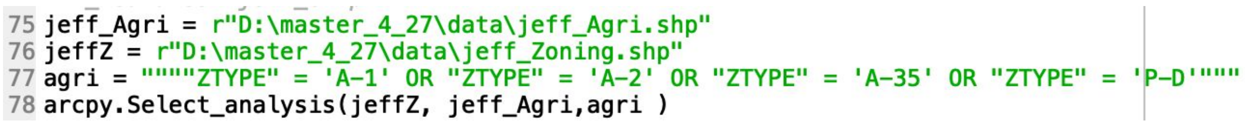
```python
with arcpy.da.UpdateCursor(jeff, ['SHAPE_AREA']) as cursor2:
    for row in cursor2:
        for area in row:
            if area < 1307000:
                cursor2.deleteRow()

del row
del area
del cursor2
```

C. Simple buffer function performed on the transmission lines (line vectors) and substation points (points).



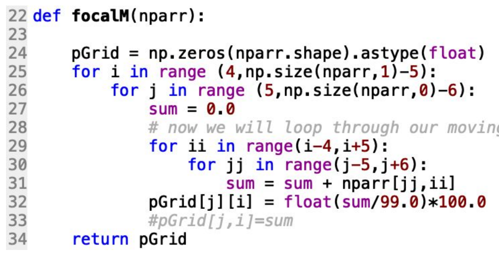
```python
lines = r"D:\master_4_23\data\CO_Electric_Power_Transmission_Lines-line.shp"#transmission lines
stations = r"D:\master_4_23\data\CO_Electric_Substations-point.shp"#substations

arcpy.Buffer_analysis(lines, r"D:\master_4_23\data\lineBuff", "3218.88 meters")
arcpy.Buffer_analysis(stations, r"D:\master_4_23\data\pointBuff", "3218.88 meters")
```

D. Jefferson County municipal zone selection; query for all agricultural and site specific zone types.



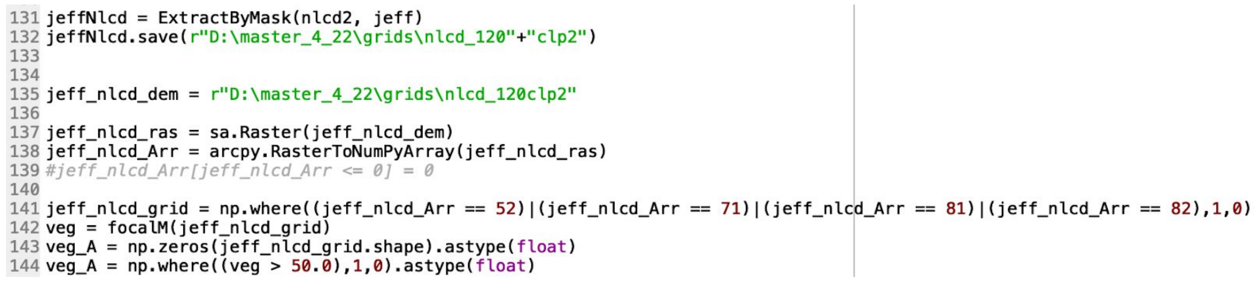
```python
jeff_Agri = r"D:\master_4_27\data\jeff_Agri.shp"
jeffZ = r"D:\master_4_27\data\jeff_Zoning.shp"
agri = """"ZTYPE" = 'A-1' OR "ZTYPE" = 'A-2' OR "ZTYPE" = 'A-35' OR "ZTYPE" = 'P-D'"""
arcpy.Select_analysis(jeffZ, jeff_Agri,agri )
```

E. Focal Mean moving window function. Acts to smooth cell values….



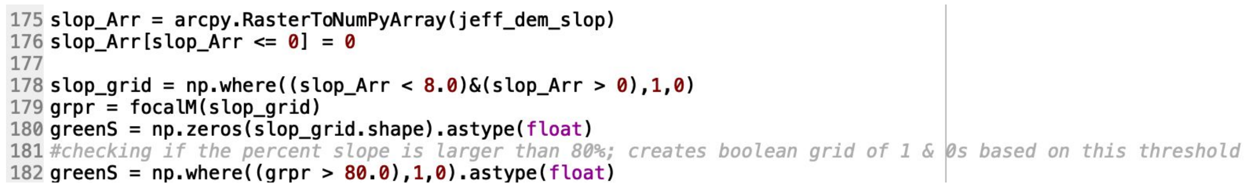
```
def focalM(nparr):

    pGrid = np.zeros(nparr.shape).astype(float)
    for i in range (4,np.size(nparr,1)-5):
        for j in range (5,np.size(nparr,0)-6):
            sum = 0.0
            # now we will loop through our moving-window
            for ii in range(i-4,i+5):
                for jj in range(j-5,j+6):
                    sum = sum + nparr[jj,ii]
            pGrid[j][i] = float(sum/99.0)*100.0
            #pGrid[j,i]=sum
    return pGrid
```

F. NLCD selection; 50% threshold



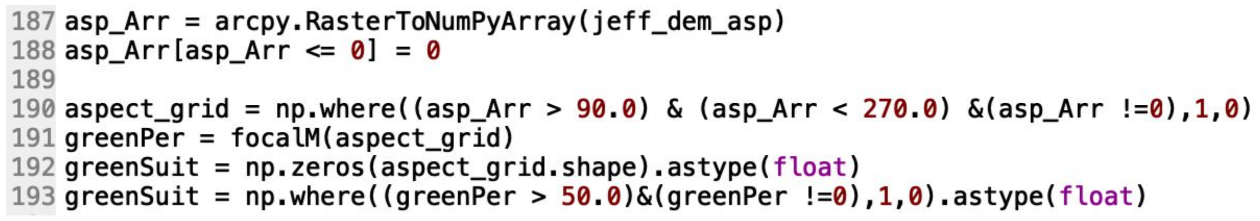
```
jeffNlcd = ExtractByMask(nlcd2, jeff)
jeffNlcd.save(r"D:\master_4_22\grids\nlcd_120"+"clp2")


jeff_nlcd_dem = r"D:\master_4_22\grids\nlcd_120clp2"

jeff_nlcd_ras = sa.Raster(jeff_nlcd_dem)
jeff_nlcd_Arr = arcpy.RasterToNumPyArray(jeff_nlcd_ras)
#jeff_nlcd_Arr[jeff_nlcd_Arr <= 0] = 0

jeff_nlcd_grid = np.where((jeff_nlcd_Arr == 52)|(jeff_nlcd_Arr == 71)|(jeff_nlcd_Arr == 81)|(jeff_nlcd_Arr == 82),1,0)
veg = focalM(jeff_nlcd_grid)
veg_A = np.zeros(jeff_nlcd_grid.shape).astype(float)
veg_A = np.where((veg > 50.0),1,0).astype(float)
```

G. Slope; 80% threshold value and slope below 8%



```
slop_Arr = arcpy.RasterToNumPyArray(jeff_dem_slop)
slop_Arr[slop_Arr <= 0] = 0

slop_grid = np.where((slop_Arr < 8.0)&(slop_Arr > 0),1,0)
grpr = focalM(slop_grid)
greenS = np.zeros(slop_grid.shape).astype(float)
#checking if the percent slope is larger than 80%; creates boolean grid of 1 & 0s based on this threshold
greenS = np.where((grpr > 80.0),1,0).astype(float)
```

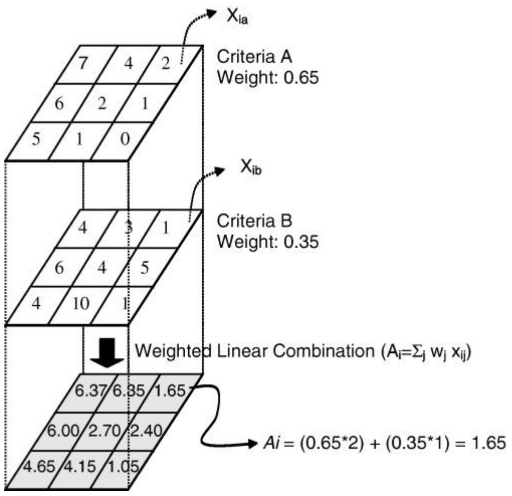
H. Aspect; 50% threshold displayed



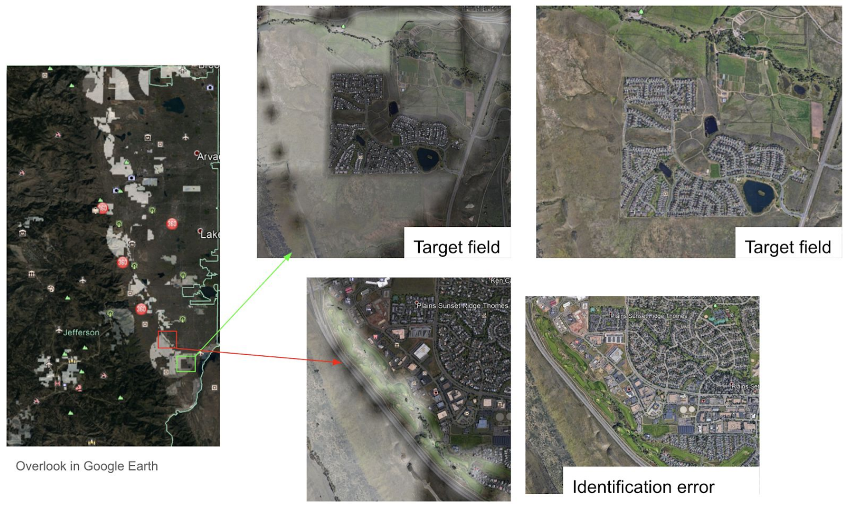
```
asp_Arr = arcpy.RasterToNumPyArray(jeff_dem_asp)
asp_Arr[asp_Arr <= 0] = 0

aspect_grid = np.where((asp_Arr > 90.0) & (asp_Arr < 270.0) &(asp_Arr !=0),1,0)
greenPer = focalM(aspect_grid)
greenSuit = np.zeros(aspect_grid.shape).astype(float)
greenSuit = np.where((greenPer > 50.0)&(greenPer !=0),1,0).astype(float)
```

I. First attempt at a cost distance function based dynamic floodplain

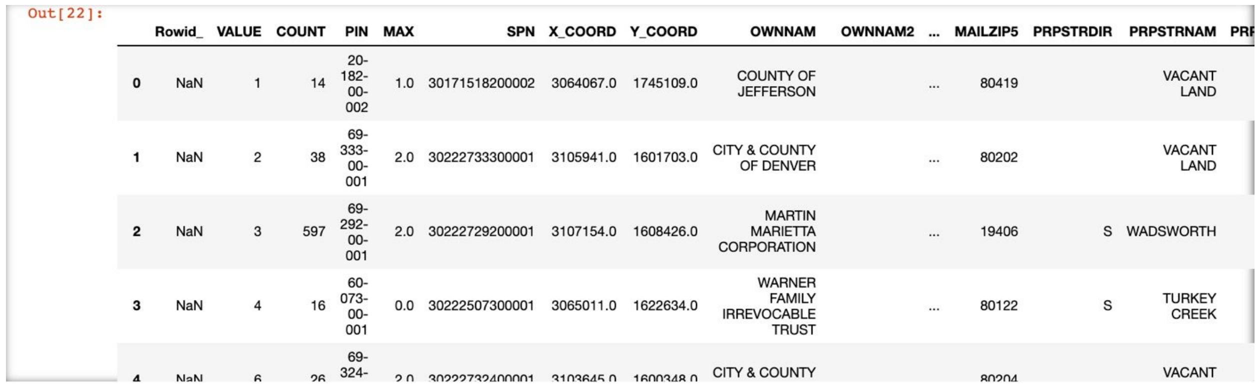
J. Graphical example of a weighted linear combination



K. Sampled output from the model. At the top right, an ideal site was correctly selected from the model. Bottom right displays a selected parcel that is currently a developed golf course. This error in selection is largely attributed to out of data data, particularly for the zoning data and NLCD data.

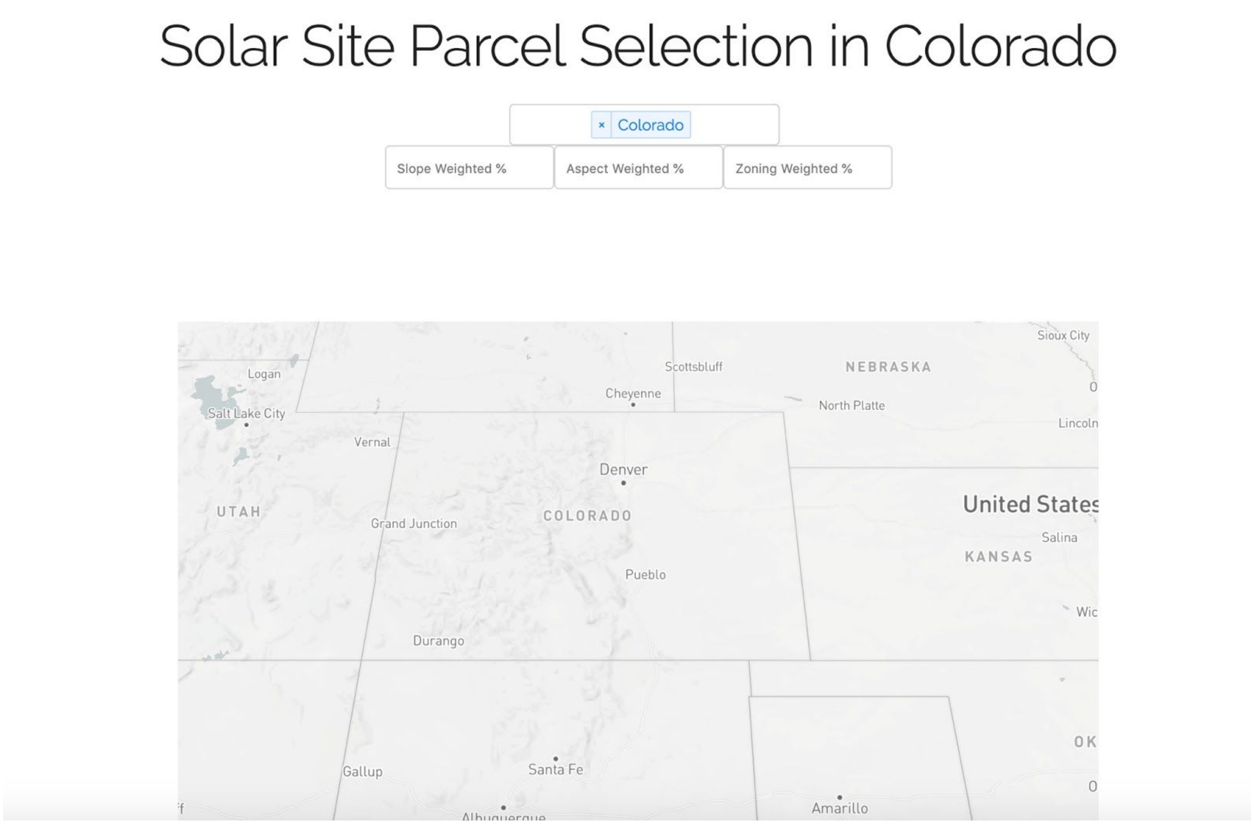


L. The top image displays the parcel information displayed in the plotly visual; the bottom image displays the mapped coordinates. These weren't displaying properly by the time the final report and code had to be submitted.



Out[22]:

	Rowid_	VALUE	COUNT	PIN	MAX	SPN	X_COORD	Y_COORD	OWNNAM	OWNNAM2	...	MAILZIP5	PRPSTRDIR	PRPSTRNAM	PR
0	NaN	1	14	20-182-00-002	1.0	30171518200002	3064067.0	1745109.0	COUNTY OF JEFFERSON		...	80419		VACANT LAND	
1	NaN	2	38	69-333-00-001	2.0	30222733300001	3105941.0	1601703.0	CITY & COUNTY OF DENVER		...	80202		VACANT LAND	
2	NaN	3	597	69-292-00-001	2.0	30222729200001	3107154.0	1608426.0	MARTIN MARIETTA CORPORATION		...	19406	S	WADSWORTH	
3	NaN	4	16	60-073-00-001	0.0	30222507300001	3065011.0	1622634.0	WARNER FAMILY IRREVOCABLE TRUST		...	80122	S	TURKEY CREEK	
4	NaN	6	26	69-324-	2.0	30222732400001	3103645.0	1600348.0	CITY & COUNTY			80204		VACANT	



Solar Site Parcel Selection in Colorado

× Colorado

Slope Weighted % | Aspect Weighted % | Zoning Weighted %