Lab 2: Cost Surfaces

Details:

1. Create ETL for data to go into cost surface model
2. Create cost surface model and justify how you created cost surface
3. Map optimal path from two points over the constructed cost surface
4. (44.127985, -92.148796) to North Picnic area (44.054388888888889 -92.04483333333333)

Specific preferences:

1. Not walk through any farm fields
2. Doesn’t like crossing water bodies if no bridge
3. Needs path that is the most gradual slopewise

Download Data

In [21]:

```python
# DEM: https://gisdata.mn.gov/dataset/elev-30m-digital-elevation-model
# Crop Area: https://gisdata.mn.gov/dataset/agri-cropland-data-layer-2018
#crop area has water on it!
import requests, json, zipfile
import json
import zipfile

def CKAN_retrieval(search_query, result_num, resource_num):

#call API to search packages with search query
    big_url = 'https://gisdata.mn.gov/api/3/action/package_search?q=' + sear

#send a request to the API address, do not verify security
    response = requests.get(big_url, verify = False)

#turn result into JSON dictionary
    json_response = json.loads(response.content)

#dig down through dictionary layers to find the right resource
    result_options = json_response['result']['results']
    chosen_result = result_options[result_num]
    resources_under_result= chosen_result['resources'][resource_num]
    chosen_resource = resources_under_result['url']
    print(chosen_resource)
    URL_request = requests.get(chosen_resource)

# define a save file name and write data to it, close file
    save_path = search_query[0:8] + ".zip"
    with open(save_path, 'wb') as f:
        f.write(URL_request.content)
        f.close()

#unzip the file into the same directory
    with zipfile.ZipFile(save_path,"r") as zip_ref:
        zip_ref.extractall()
    print('Complete')

#function calls
CKAN_retrieval('elev-30m-digital-elevation-model',0,1)
CKAN_retrieval('agri-cropland-data-layer-2018',0,1)
```

```
C:\Program Files\ArcGIS\Pro\bin\Python\envs\arcgispro-py3\lib\site-packages\url
lib3\connectionpool.py:1004: InsecureRequestWarning: Unverified HTTPS request i
s being made to host 'gisdata.mn.gov'. Adding certificate verification is stron
gly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#
ssl-warnings (https://urllib3.readthedocs.io/en/latest/advanced-usage.html#ssl-
warnings)
  InsecureRequestWarning,

https://resources.gisdata.mn.gov/pub/gdrs/data/pub/us_mn_state_dnr/elev_30m_dig
ital_elevation_model/fgdb_elev_30m_digital_elevation_model.zip (https://resourc
es.gisdata.mn.gov/pub/gdrs/data/pub/us_mn_state_dnr/elev_30m_digital_elevation_
model/fgdb_elev_30m_digital_elevation_model.zip)
Complete. Check notebook folder

C:\Program Files\ArcGIS\Pro\bin\Python\envs\arcgispro-py3\lib\site-packages
```

```
\urllib3\connectionpool.py:1004: InsecureRequestWarning: Unverified HTTPS re
quest is being made to host 'gisdata.mn.gov'. Adding certificate verificatio
n is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advance
d-usage.html#ssl-warnings (https://urllib3.readthedocs.io/en/latest/advanced
-usage.html#ssl-warnings)
  InsecureRequestWarning,

https://resources.gisdata.mn.gov/pub/gdrs/data/pub/us_mn_state_mda/agri_croplan
d_data_layer_2018/fgdb_agri_cropland_data_layer_2018.zip (https://resources.gis
data.mn.gov/pub/gdrs/data/pub/us_mn_state_mda/agri_cropland_data_layer_2018/fgd
b_agri_cropland_data_layer_2018.zip)
Complete. Check notebook folder
```

Module Imports

In [1]:

```python
#model builder version

import arcpy
from arcpy.sa import *

# Check out any necessary licenses.
arcpy.CheckOutExtension("3D")
arcpy.CheckOutExtension("spatial")
arcpy.CheckOutExtension("ImageAnalyst")

Lab2_gdb = "C:\\Users\\Cole\\Documents\\GitHub\\GIS5572\\Lab2\\Lab2.gdb"
agri_cropland_data_layer_2018 = arcpy.Raster("C:\\Users\\Cole\\Documents\\Gi
digital_elevation_model_30m = arcpy.Raster("C:\\Users\\Cole\\Documents\\GitH
```

Reclassify

In [9]:

```
# Process: Reclassify Ag/Water
#1 = acceptable, 100 not acceptable
Rec_AG_WTR = "C:\\Users\\Cole\\Documents\\GitHub\\GIS5572\\Lab2\\Lab2.gdb\\R
arcpy.ddd.Reclassify(in_raster=agri_cropland_data_layer_2018, reclass_field=
Rec_AG_WTR = arcpy.Raster(Rec_AG_WTR)

# Process: Convert DEM to Slope
Output_raster = "C:\\Users\\Cole\\Documents\\GitHub\\GIS5572\\Lab2\\Lab2.gdb
arcpy.ddd.Slope(in_raster=digital_elevation_model_30m, out_raster=Output_ras
Output_raster = arcpy.Raster(Output_raster)

# Process: Rescale Slope by Function
#steeper = larger value
Resc_SLOPE = "C:\\Users\\Cole\\Documents\\GitHub\\GIS5572\\Lab2\\Lab2.gdb\\R
Rescale_by_Function = Resc_SLOPE
Resc_SLOPE = arcpy.sa.RescaleByFunction(in_raster=Output_raster, transformat
Resc_SLOPE.save(Rescale_by_Function)

# Process: Add slope and ag/water rasters together to create surface
Surface = "c:\\Users\\Cole\\documents\\GitHub\\GIS5572\\Lab2\\Lab2.gdb\\Surf
Raster_Calculator = Surface
Surface =  Rec_AG_WTR+ Resc_SLOPE
Surface.save(Raster_Calculator)
```

Clip

In [10]:

```
#Clip Raster
import arcpy

# define a clip extent
clip = "560000 4850000 600000 4900000"

# clip each layer by clip extent and save output to Clip_xxx
arcpy.Clip_management("Lab2.gdb\\Surface", clip, "Lab2.gdb\\Clip_Surface")
```

Out[10]:

Output

Lab2.gdb\Clip_Surface

Messages

Start Time: Wednesday, March 3, 2021 11:06:42 AM

Building Pyramids...

Succeeded at Wednesday, March 3, 2021 11:06:43 AM (Elapsed Time: 0.92 seconds)

Create Source and Destination Feature Layers

In [11]:

```
#gather the spatial referenence from the cost surface
spatial_ref = arcpy.Describe("Lab2.gdb\\Clip_Surface").spatialReference

#create a new feature class for the source point
arcpy.CreateFeatureclass_management("Lab2.gdb", "SourceModel", "POINT",
                                    spatial_reference = spatial_ref)

#create a new feature class for the dest point
arcpy.CreateFeatureclass_management("Lab2.gdb", "DestModel", "POINT",
                                    spatial_reference = spatial_ref)

#add a point to the new source feature class
feature_class_sour = "Lab2.gdb\\SourceModel"
cursor = arcpy.da.InsertCursor(feature_class_sour,"SHAPE@XY")
xy = arcpy.Point(568097.73, 4886440.22)
cursor.insertRow([xy])
del cursor

#add a point to the new dest feature class
feature_class_dest = "Lab2.gdb\\DestModel"
cursor = arcpy.da.InsertCursor(feature_class_dest,"SHAPE@XY")
xy = arcpy.Point(576512.44, 4878357.35)
cursor.insertRow([xy])
del cursor
```

Distance Accumulation

In [12]:

```
# Process: Distance Accumulation (Distance Accumulation) source >> points on
arcpy.env.workspace = "C:\\Users\\Cole\\Documents\\GitHub\\GIS5572\\Lab2\\La
AccumSurface = arcpy.sa.DistanceAccumulation(in_source_data="SourceModel",
                              in_barrier_data="",
                              in_surface_raster="",
                              in_cost_raster="Clip_Surface",
                              out_back_direction_raster="BackRaster",
                              source_cost_multiplier="",
                              distance_method="PLANAR")
AccumSurface.save("AccumSurface")
BackRaster = arcpy.Raster("BackRaster")
```

Optimal Path

In [13]:

```
# Process: Optimal Path As Line
# distance accum >> destination
arcpy.env.workspace =  "C:\\Users\\Cole\\Documents\\GitHub\\GIS5572\\Lab2\\L
Path = arcpy.sa.OptimalPathAsLine(in_destination_data="DestModel",
                            in_distance_accumulation_raster= "AccumSurface",
                            in_back_direction_raster=BackRaster,
                            out_polyline_features= "OutPolyline",
                            destination_field="ObjectID",
                            path_type="BEST_SINGLE")
#Path.save("Lab2.gdb\\OutPolyline")
```