# Overview:

[cleanup\_costlines\_biomass\_v1.py](file:///M:\Spatially_explicit\01_Projects\21_BC_wildfires\MC_Fire\05_working\spatial_analysis\bioenergy\cleanup_costlines_biomass_v1.py) is a script that joins Nick’s original costline outputs, which are split by year because of Network Analyst size constraints (copy in /[bioenergy](file:///M:\Spatially_explicit\01_Projects\21_BC_wildfires\MC_Fire\05_working\spatial_analysis\bioenergy\cleanup_costlines_biomass.py) ; EventODCostlinesCarbonTonnes.gdb), and merges them with the yearly HarvCent point features (copy in in /[bioenergy](file:///M:\Spatially_explicit\01_Projects\21_BC_wildfires\MC_Fire\05_working\spatial_analysis\bioenergy\cleanup_costlines_biomass.py); BiomassCarbonTonnesBCAlbers.gdb), based on the same file suffix (e.g., 031 represents year 31, i.e., 2020, thus these two files would be matched).

This script is useful to clean up Nick’s output so that the paved and unpaved distances are associated with each costpoint. Its also useful to input into the bioenergy optimization tools.

The script should be run from the same directory that the two geodatabases are stored in. For example, if the two geodatabases are stored in /bioenergy, then the script should be run from /bioenergy. No other setup is needed.

# Steps:

1. Create range of the filenames to be input so the year range of input files can be flexible (e.g., for testing purposes; lines 45-55).
2. Create blank list for the final spatial file to collect intermediate data layers in (line 57)
3. Add new fieldnames specifying the x,y origin and destination of the costlines (line 62-80)
   1. Line 65-70 adds the 4 new blank fields
   2. Line 73-80 fills in the fields with the x,y origin and endpoints
4. Convert costline to originpoint based on the x,y origin point (lines 81-111)
   1. Line 85-88 converts the line file to a table
   2. Line 91- 111 converts the table file to a point based on the costline origin coordinates
5. Make a unique ID for each costline\_point based on the unique ID it already has plus the year number in the filename (line 112- 141)
   1. This makes a new field based on NEAR\_FID and Year, so each new id will be identified based on the closes t road feature for that given year. It may be that some features are close to the road feature so the ID may appear twice. In that case, the final output first column is a unique object ID that can also be used.
6. Create a unique tc field based on the input tc/ha field (line 142- 165)
   1. Line 146 converts the latitude and longitude coordinates to meters (UTM)
   2. Line 160 does the field calculation based on the latitude field
7. Create the summed rank fields and reduce to one point per spatial location (so that all 20 ranks are in one row) (line 171- 350)
   1. Line 171- 196 contains a function to easily construct the names without repeating them manually and defines a function for reading the names of the fields using an indexing function so that the code is easier to interpret later on
   2. Line 199-217 adds in blank fields based on the ones constructed above (for rank 1-20,it adds cost, length, and type of transportation)
   3. Line 221- 256 collects the new field names from the output file and adds in the corresponding sums for each cost, length, and type of transport (.e.g, paved adds with cityspoke to give Total\_Paved)
   4. Build the names of the fields to be dissolved with the respective type of dissolve to perform (e.g., rank4\_dest MAX;) (line 258-284)
   5. One the fields are renamed in the dissolve operation, they have to be converted back to regular names (e.g., rank4\_dest\_MAX becomes rank4\_dest again) by using a dictionary matching/alter field operation (line 289-345)
8. Merge the costline with the costpoint based on the year number (line 353-389)
   1. First make a dictionary based on the years matching (line 354-376)
   2. Then accumulate all the features together using a spatial join (spatially joins year by year each costpoint file to each costline file)
9. Perform the near analysis (line 396-412)
   1. This isn’t really necessary because Nick already processed the near features and they were correct. It may be useful for different input data. In that case you can uncomment line 412.
10. Convert the outputs into a temporary .dbf in order to reproject as points along the road network (line 414- 438)
11. Merge as a final spatial table from individual spatial output layers (line 441-448)
12. Output aspatial table (line 451) (if the process breaks on this final line, outputusing table to table too, ensuring that the output name contains “.csv” in a non-geodatabase directory).

**The following lines can be modified in order to change the number of years and ranks being processed:**

**Line 46**- changes the total number of years to process

e.g., can change the max year from 81 (2017) to 33 (2022).

**Line 172/227/270-** changes the number of ranks to generate in the final output

e.g., can change the max rank from 20 to 3.

***Note: lines 227 and 270 must be written as (rank you want)+1. So if you want ranks 1-3, the script should have 1-4 instead. Line 172 is not affected by this rule.***