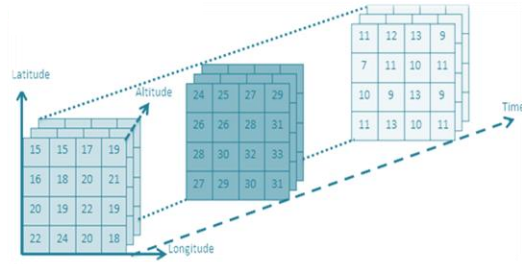
Multidimensional Data[1]



The ability to take data - to be able to understand it, to process it, to extract value from it, to visualize it, to communicate it is going to be a hugely important skill in the next decades. – Hal Varian, 2009
Written by Sumeeta Srinivasan Updated November 2025

- Working with Lidar data (3D with Z)
- Working with Netcdf (3D with attributes)
- Animation using the Time toolbar (4D)
- Working with multivariate data (nD) in grouping and principal component analysis
- Using Python for iteration

Estimated Time: 4-5 hours

Contents

Working with data in many dimensions

In this exercise you will work with adding more dimensions to data besides the X and Y which you use to locate a coordinate in two-dimensional space. You have worked with the third dimension or Z as in elevation or as time in previous tutorials. You will now also work with classifying data that can be thought of as n dimensional (many attributes) so that you can look for patterns (signals) amongst all the data (noise).

Data for this lab exercise

The three datasets that you will use in this tutorial are:

[1] *Air temperature represented as a 4D hypercube courtesy:* ***geoserver.geo-solutions.it/edu/en/multidim/intro/intro.html***

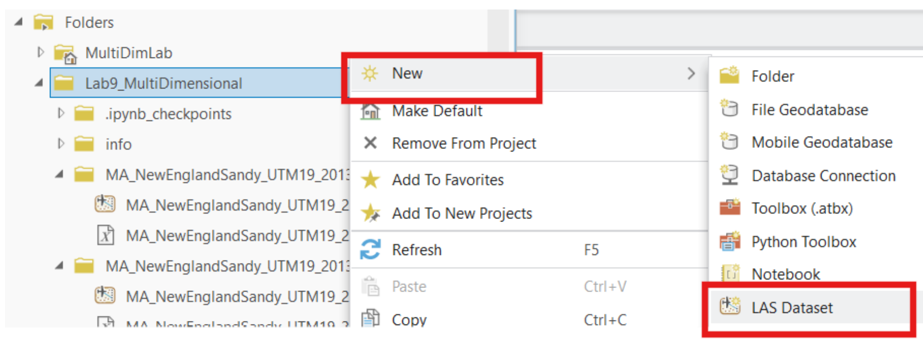
- a **las** data file with Lidar data for part of the Tufts Medford/ Somerville campus in the las format
- a **netcdf** file gpcp_agg_1979_2014.nc which has precipitation data for the world from 1979-2014
- Environmental performance index data by country for 2022 in a **gdb** called EPI.gdb

1. Copy the data into your own folder for this exercise.
2. Open ArcGIS Pro with a new project in your folder for this exercise

1.0 Lidar Data

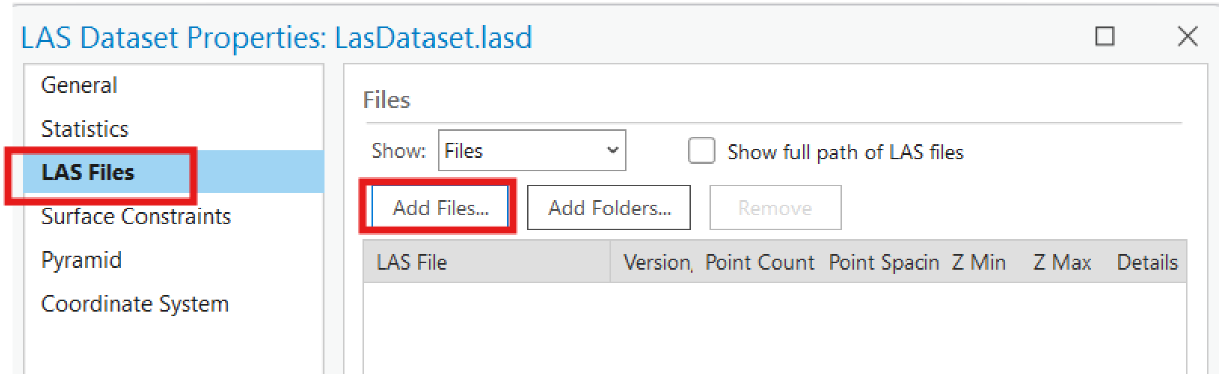
ArcGIS Pro can work directly with LAS data using the LAS Dataset format. LAS data is not visible in Catalog outside of the LAS Dataset (lasd). LAS data can be downloaded at the USGS through the Lidar explorer: https://apps.nationalmap.gov/lidar-explorer/. Lidar data are distributed as laz files that need to be unzipped using a utility such as LASzip[2]. As described by the USGS "The LAS format is a standardized binary format for storing 3-dimensional point cloud data and point attributes along with header information and variable length records specific to the data. Millions of data points are stored as a 3-dimensional data cloud as a series of geo-referenced x, y coordinates and z (elevation), as well as other attributes for each point."

3. In ArcGIS go to the **Catalog** tab and browse to your own folder for this exercise. Right click on your folder (not the gdb) and select **New, LAS Dataset.**

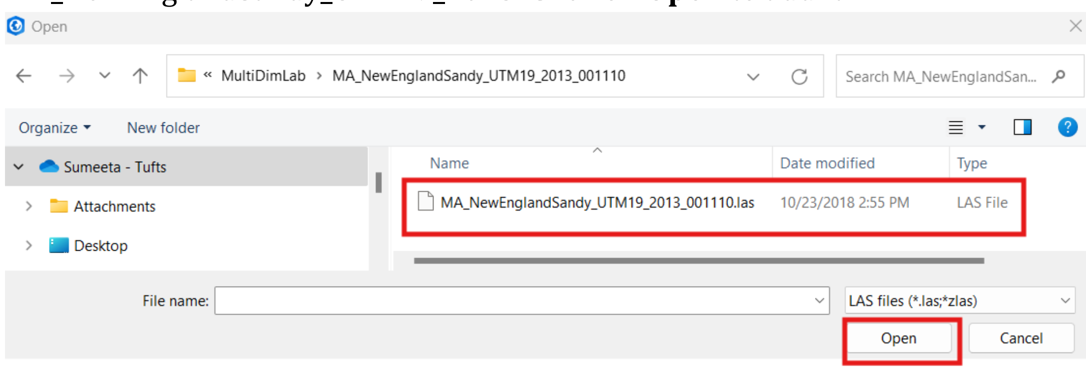


4. Rename the new lasd dataset to be **Tufts.lasd.**
5. Right click on Tufts.lasd in the **Catalog** (**not Contents**) and select **Properties** to open a new window.
6. Click on the tab for LAS **Files**.

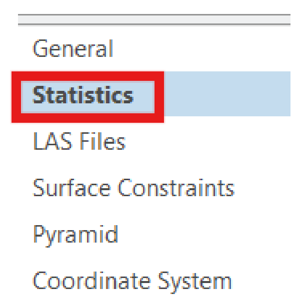
[2] https://laszip.org/



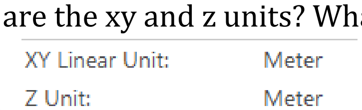
7. Click on **Add files** in the **LAS Files** tab and browse to the folder where you have data for this lab and select it from one of the folders which begin with MA_NewEnglandSandy_UTM19_2013. Click on **Open** to add it in.



Note that if you have many las files you should use the Shift key to select multiple las files.

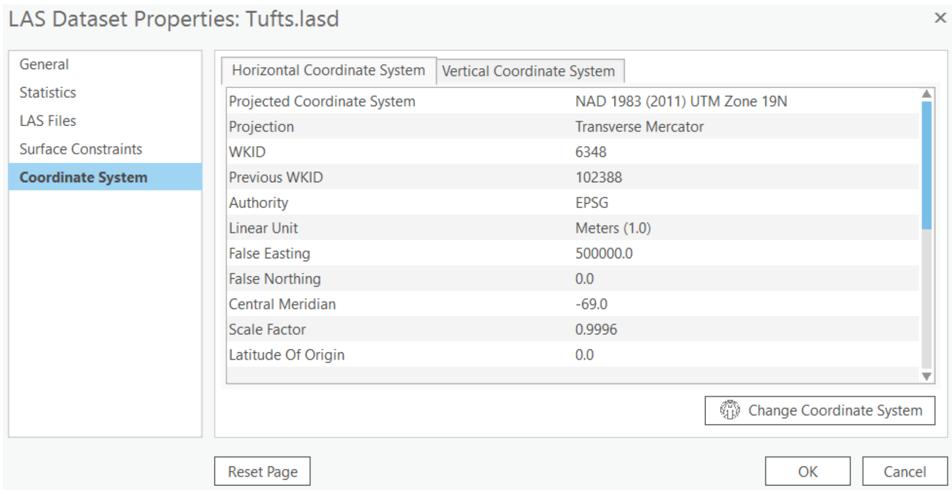


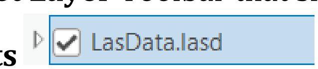
8. Click on the **Statistics** tab. If it is empty, click on **Update**. Note that for even these relatively small tiles (roughly 1mile x 1 mile) there are many millions of points (the point count)!
9. Click on the **General** Tab. What are the xy and z units? What is the highest measured z in this data?

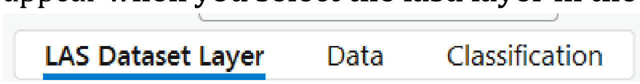


10. Click on the **Coordinate System** tab.

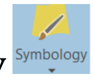
What is the projection, coordinate system for this data? Notice it also has a Vertical Coordinate System



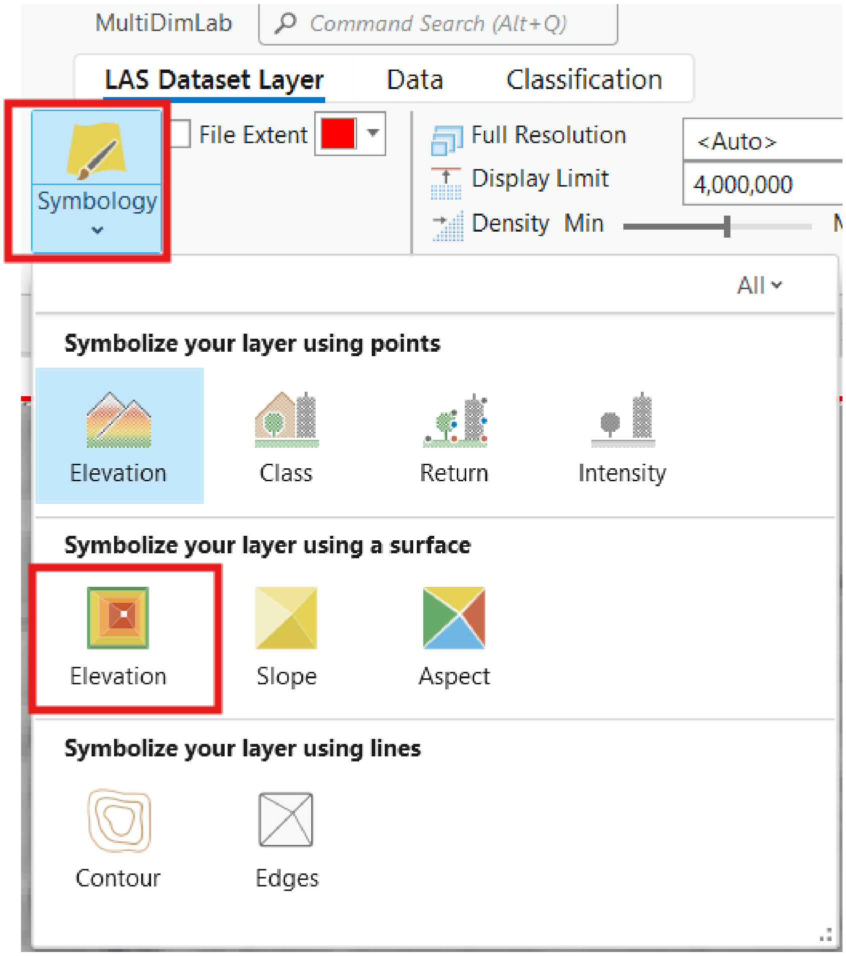
11. Click on **OK** to dismiss the properties window.
12. Drag the now configured LAS Dataset from the Catalog window into the main Map window. (Click okay if it asks to build a dataset Pyramid[3]).
13. To work with the LAS Dataset, you need the **LAS Dataset Layer** Toolbar that should appear **when you select the lasd layer in the Contents** 



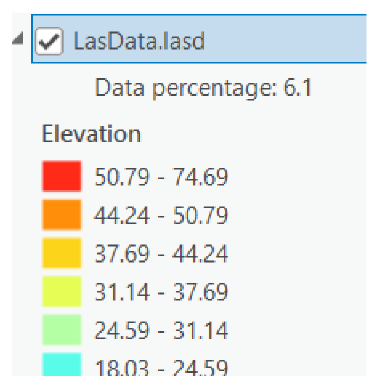
[3] https://pro.arcgis.com/en/pro-app/latest/tool-reference/data-management/build-pyramids.htm

14. In the **LAS Dataset Layer** pane, click on **Symbology** 

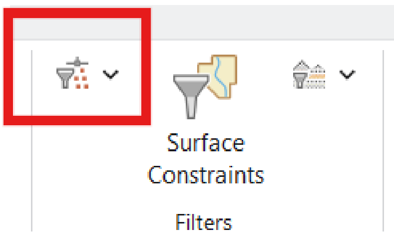
and select **Symbolize your layer using a surface and select Elevation.**

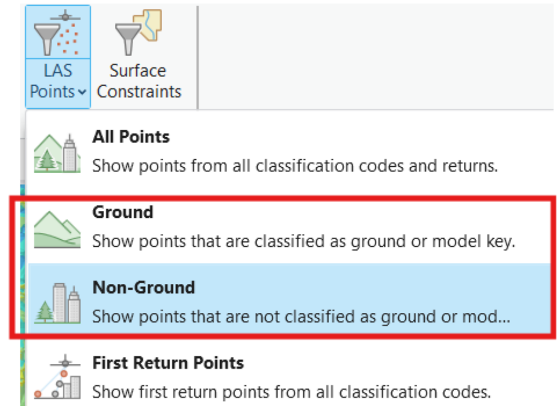


15. Look at the layer. It shows the colors assigned to each elevation range and the percentage of data being used for the display. **Zooming in** will increase the percentage of data used in the display.



16. Click on LAS Points



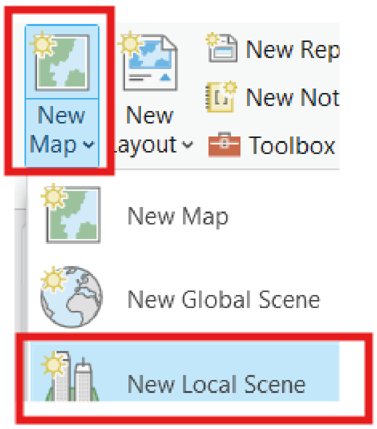


17. Select **Ground** to see the elevation of the ground.
18. Then select **Non ground** shows you the buildings as well as ground.

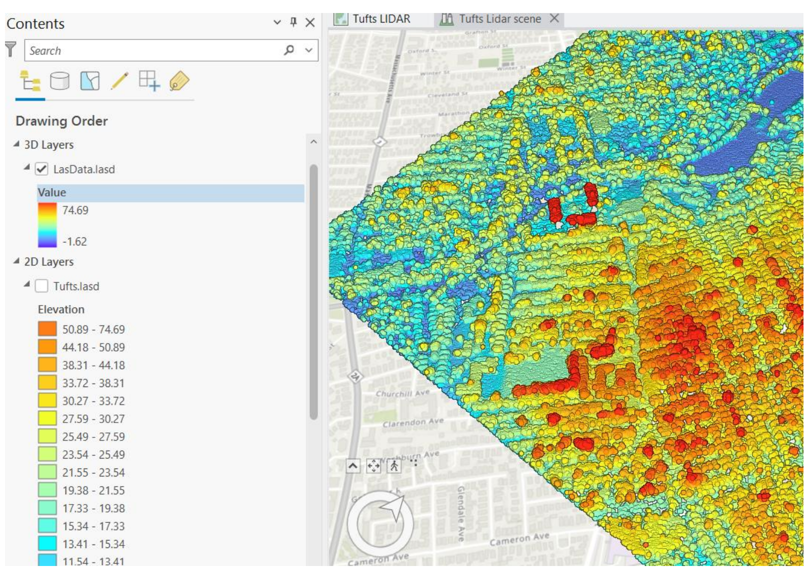
Notice how different they look.

19. Insert a **New local scene** from the **Insert** pane 

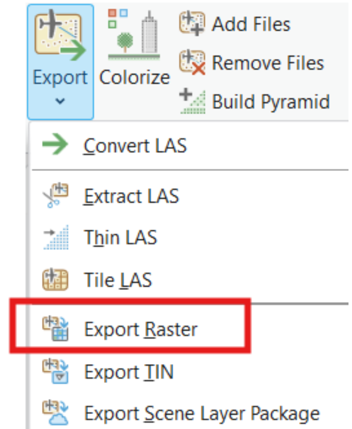
, **New Map**.



20. Drag in the lasd file from the Catalog into this scene. You can visualize the data in 2D and 3D.



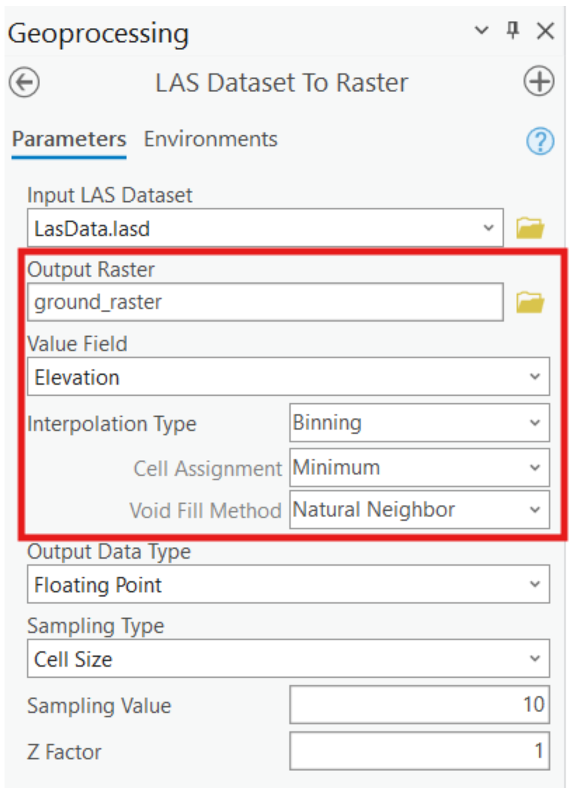
21. Go back to the **map** with the lidar data (not the local **scene**)



22. In the **Data** pane 

(make sure your LAS data layer is highlighted in the Contents) click on **Export** and then select **Export Raster**

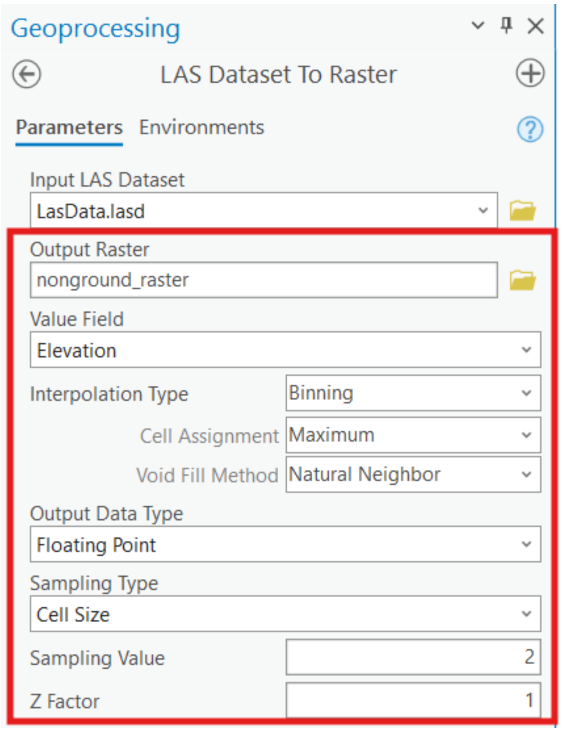
23. Using the ground LAS points you can create a ground or bare-earth raster. This is the Digital Terrain Model or the elevation at the ground level including natural features.

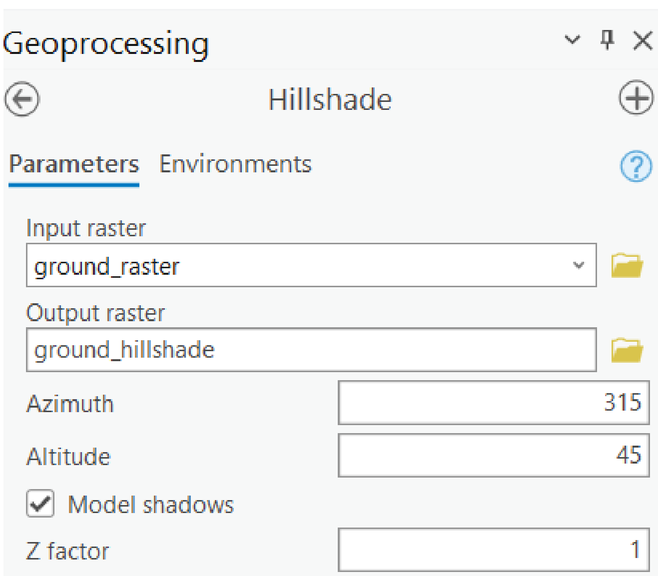


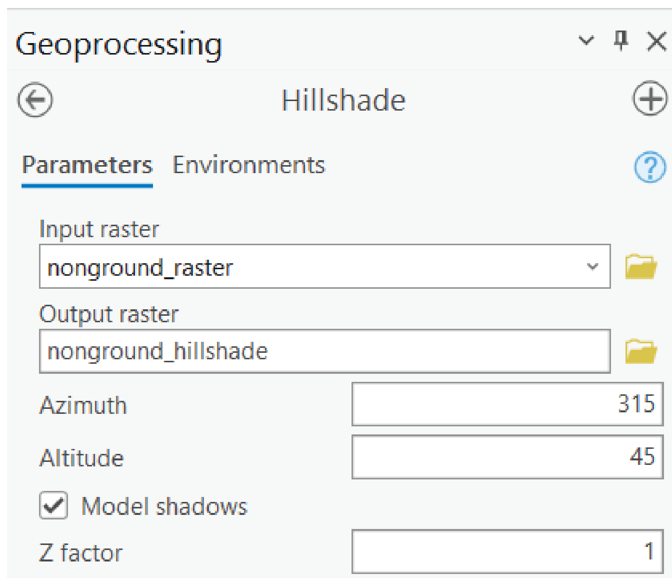
24. Create a hillshade from this DTM as shown in the screenshots. (Use the hillshade geoprocessing tool). See screenshots that follow.

25. Repeat the steps with the LAS points that are non-ground 

to get the DSM. Notice that I made my cell size fine-grained (2 m) since buildings can be pretty narrow.

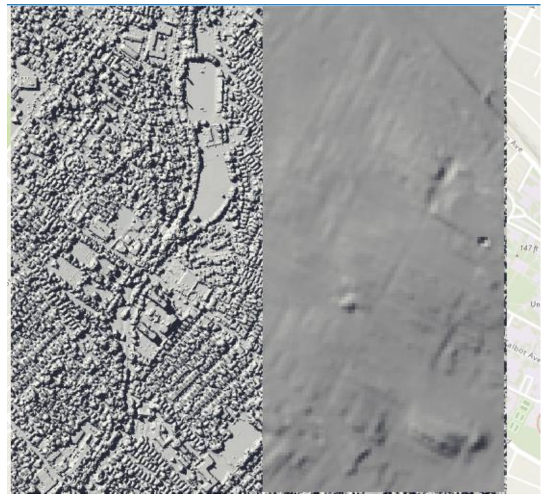






In the Raster Layer tab pane click on Swipe to select the DSM layer and move the cursor onto the map and click once and allow the cursor to return to an arrowhead.

26. Click and hold dragging the cursor down and revealing the DTM underneath the DSM



Part 1 Question (1 point)

Attach screenshots showing both the DTM and DSM

2.0 Grouping and Dimension Reduction analysis

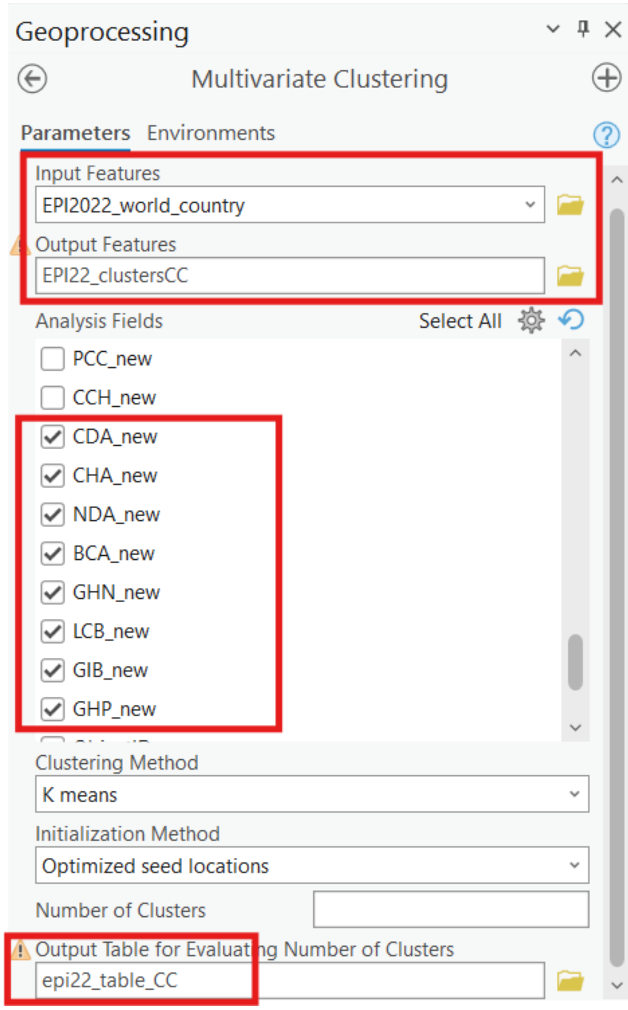
You can think of a data layer with many attributes as data that are multidimensional or n dimensional. This could be data over many years or just many attributes associated with the place. For this section you will use EPI or environmental performance index data. The authors of the dataset " The 2022 Environmental Performance Index (EPI) provides a data-driven summary of the state of sustainability around the world. Using 40 performance indicators across 11 issue categories, the EPI ranks 180 countries on climate change performance, environmental health, and ecosystem vitality. These indicators provide a gauge at a national scale of how close countries are to established environmental policy targets." The variables are described in the Excel file: epi2022.xlsx.

When you have many hundreds of variables to describe countries over time it might be hard to see patterns. ArcGIS notes that "Whenever we look at the world around us, it is very natural for us to organize, group, differentiate, and catalog what we see to help us make better sense of it; this type of mental classification process is fundamental to learning and comprehension. Given the number of clusters to create, it will look for a solution where all the features within each cluster are as similar as possible, and all the clusters themselves are as different as possible. Feature similarity is based on the set of attributes that you specify for the **Analysis Fields** parameter, and clusters are created using the K-Means algorithm." The goal of the K Means algorithm is to partition features so the differences among the features in a group, over all groups, are minimized. Because the algorithm is **NP-hard**, a greedy heuristic is employed to group features. The greedy algorithm will always converge to a local minimum but will not always find the global (most optimal) minimum.

1. Insert a new map in your project for this part of the exercise.

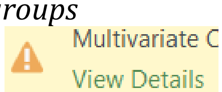
2. In the Analysis Tools, navigate to Spatial Statistics Tools, Mapping Clusters, **Multivariate Analysis** to get a new window.
3. In the window for the tool use EPI2022_world_country (which is a feature class in in EPI.gdb) as the input feature layer, select about 5-10 variables that you will use for your grouping. I used K means as the clustering method. Make sure that you save the output table for evaluating clusters. (See screenshot)

   The help notes that "While there is a tendency to want to include as many Analysis Fields as possible, for Multivariate Analysis, it works best to start with a single variable and build. Results are easier to interpret with fewer analysis fields. It is also easier to determine which variables are the best discriminators when there are fewer fields"



The first run will give you a sense of the optimal number of groups if you leave the number of clusters blank.

Note: *Even if you have errors saving the resulting table the messages will show you the optimal number of groups*

4. Click on  to see the messages.

   In the case of one of my runs, the highest pseudo F statistic was for 3 clusters.

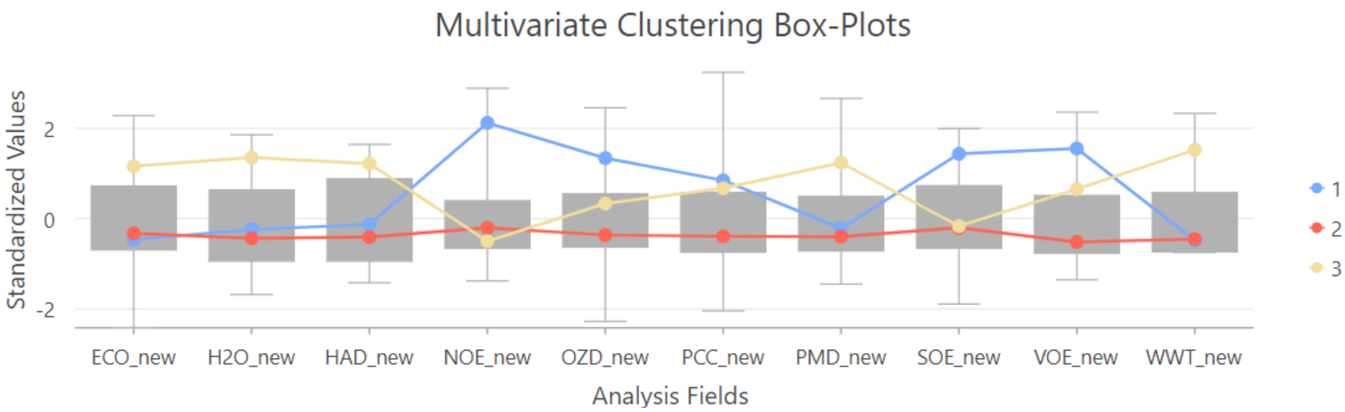
   The R^2 value in the table in the messages reflects how much of the variation in the original data was retained after the grouping process, so the larger the R^2 value is for a variable, the better that variable is at discriminating among your features. Thus WWT (Waste water treatment) was better than ECO (Ecosystem vitality) in differentiating between countries.

NUM_GROUPS	PSEUDO_F
3	76.072307
2	73.775077
4	71.911926
5	63.711074
6	57.441295
7	52.151234

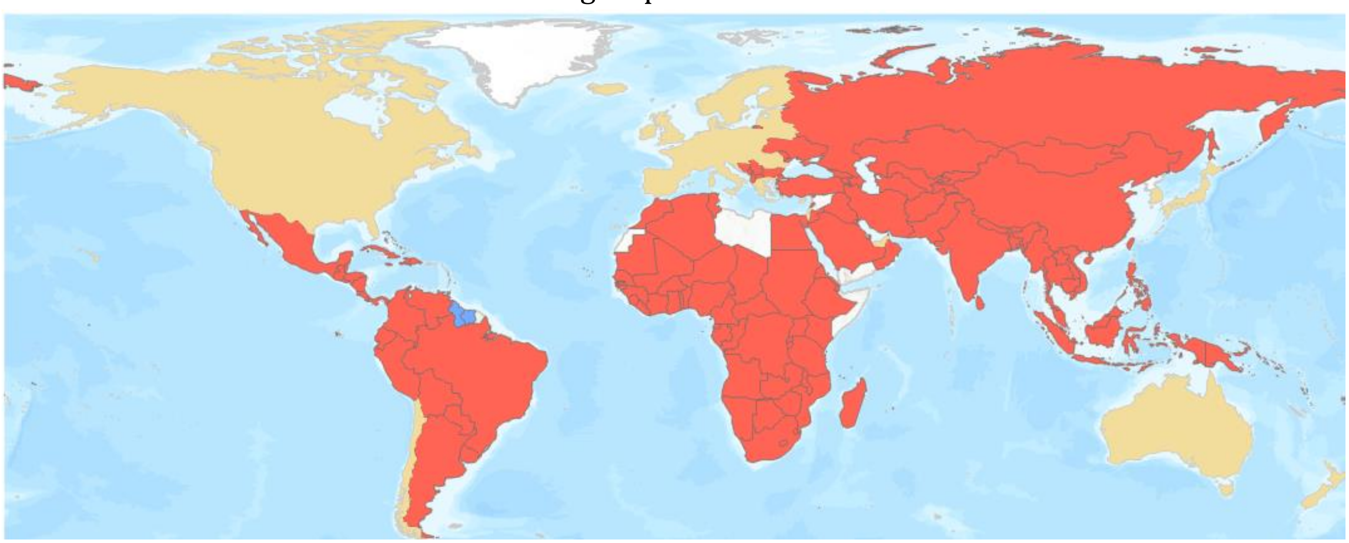
Variable-Wise Summary

Variable	Mean	Std. Dev.	Min	Max	R2
WWT_NEW	24.390395	32.395289	0.000000	100.000000	0.700124
HLT_NEW	42.937853	21.614835	10.900000	94.700000	0.636949
H2O_NEW	47.559887	28.229988	0.000000	100.000000	0.554021
PMD_NEW	35.291525	24.271689	0.000000	100.000000	0.461421
ECO_NEW	45.115819	12.588102	14.600000	73.900000	0.407370

The resulting layer comes with some useful charts that help you understand the data a bit more. For example, in one of my runs I got this chart:



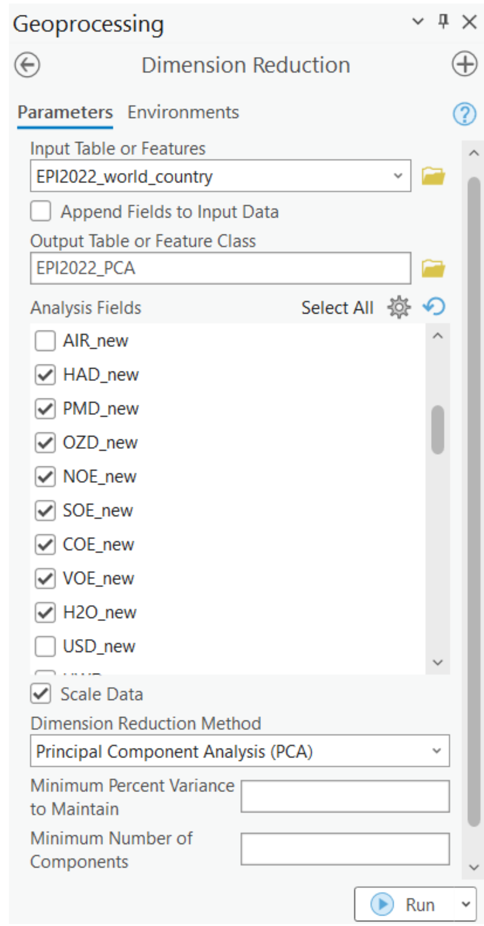
This suggests that the red group has below average index values for all the variables I chose. This groups included most of Asia, Africa and South America as seen in the screenshot of the map. High values were scored by European and North American countries. Guyana and Surinam (the blue group) scored well on the air quality indices and were thus very different from the other countries in both the other groups.



Note that I used the world boundaries feature class as a basemap because some countries did not have data for the EPI index.

Another way to make sense of a lot of variables is to use Dimension reduction with Principal Components analysis (PCA). ArcGIS's help about PCA:

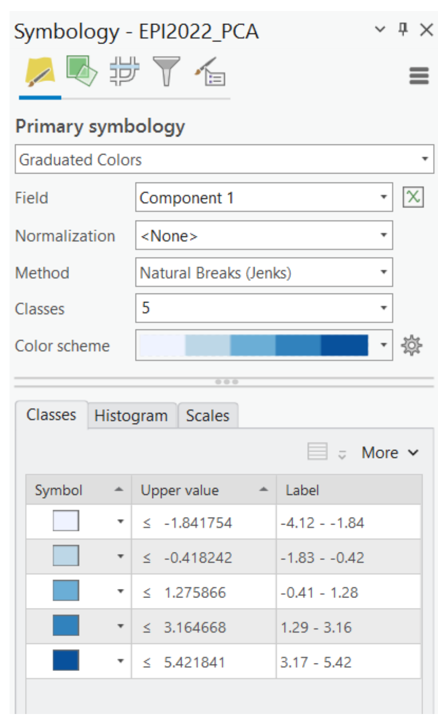
This method sequentially builds components that each capture as much of the total variance and correlations between the original variables as possible. The **Scale Data** parameter can be used to scale each original variable iso that each variable is given equal importance in the principal components. If the data is not scaled, variables with larger values will account for most of the total variance and will be overrepresented in the first several components. This method is recommended when you intend to perform an analysis or machine learning method in which the components are used to predict the value of a continuous variable. PCA works by sequentially building components that each capture a certain percent of the total variance of all of the analysis fields. Each component itself is a linear combination (weighted sum) of each of the analysis fields, where the weights are called the loadings of the component.



5. Use the same variables from your cluster analysis in the Dimension reduction tool. Search for it in the Analysis tools

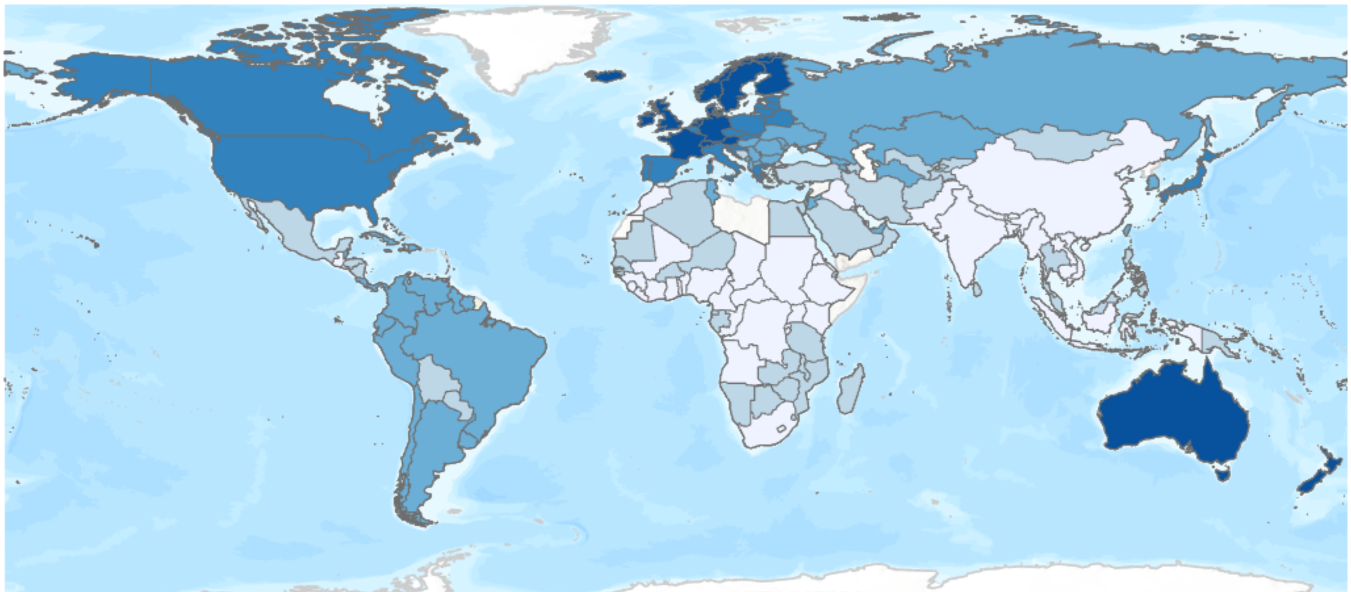
6. Symbolize the Component 1 for this as shown in the screenshot.

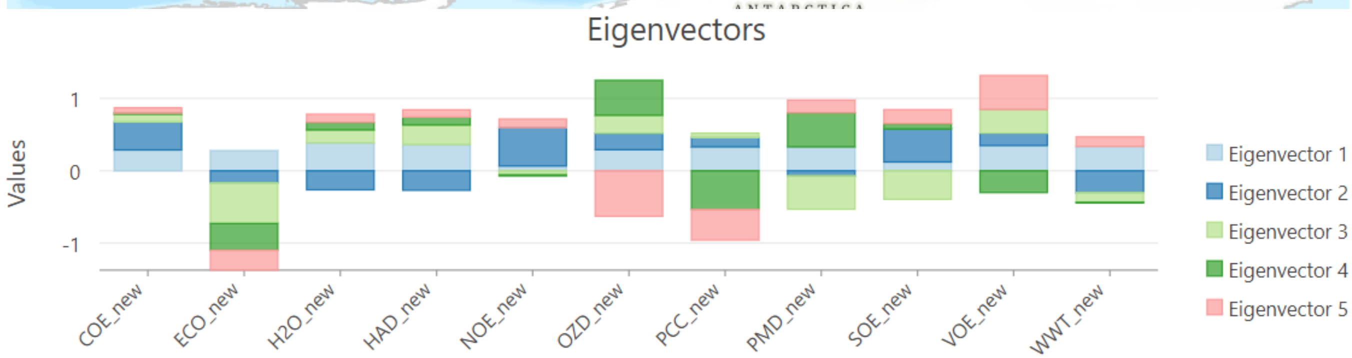
The first component explains the highest amount of the variance in the data. It is therefore useful to map the first component. In this case a look at the eigenvector (make sure to save this as a table) suggests that this component has high values for countries good air and water quality with the highest values in western Europe.



Eigenvalues

Component	Percent of Variance Explained	Cumulative Percent of Variance Explained
1*	39.47	39.47
2*	28.49	67.96





Part 2 Question (2 points)

Attach the map of clusters, the automatically generated box plot, and an explanation of the grouping that you found for a set of variables of your choice from the EPI data. Use Dimension reduction (PCA) with the same data and attach a screenshot of the first component. What does the map appear to indicate? What patterns do you see? How is it different (or similar) from the clusters from the Cluster map?

3.0 More Python, more code, cursors

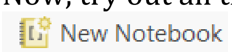
Geoprocessing tasks may require record by record access of field values. This is done in ArcPy by using an object that points to a record in a table or feature class called “**cursor**” (in the data access

module). Cursors are commonly used to read and update attributes. There use is highly recommended due to high performance and easy operation.
There are three types of cursors:

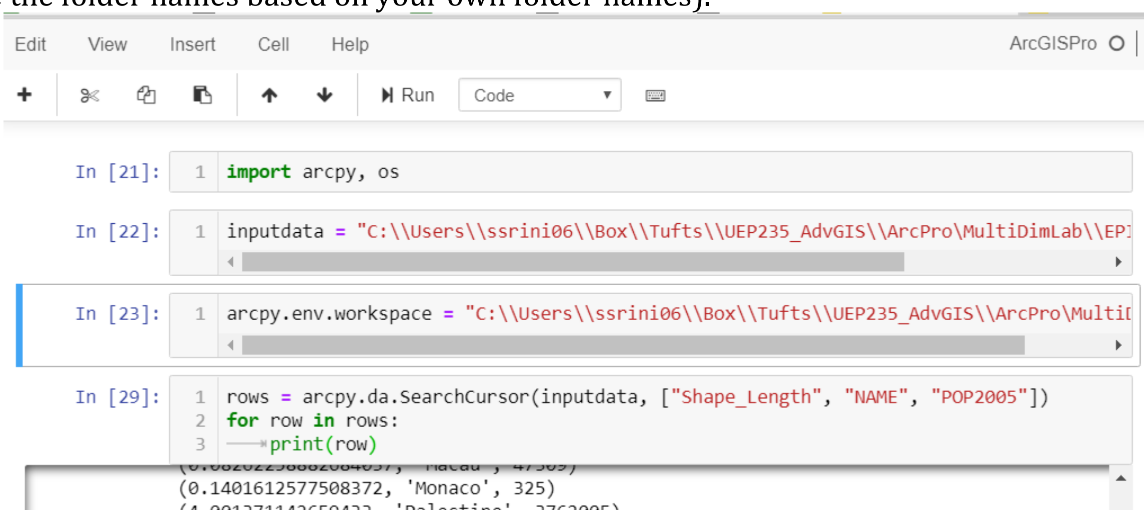
- **Search Cursor**
  Read-only access
- **Update Cursor**
  Read/Write/Delete access but can't create new records
- **Insert Cursor**
  Read/Write access with capability of creating new records

A row object is returned from the search cursor object
• Fields are accessed as properties of the row object
• The row object's GetValue and SetValue methods can be used if the field name is a variable
• The row and cursor objects need to be "destroyed" to remove read locks on the data source

The syntax of SearchCursor is:
arcpy.da.searchCursor (in_table, field_names, {where_clause}, {spatial_reference}, {explode_to_points}, {sql_clause})

Now, try out all the cursor examples in the notebook Cursor_tests. First insert a new notebook 

from the Insert menu and then test the code as shown below. (Make sure to change the folder names based on your own folder names).



Do not copy and paste from the pdf document – use the .py text file to copy and paste from or use the notebook in the project file.

```
# change folder path to your own folder for this lab
import arcpy
inputdata =
"C:\\Users\\ssrini06\\Box\\Tufts\\UEP235_AdvGIS\\ArcPro\MultiDimLab\\EPI.gdb\\world
_country_poly"

#Open a SearchCursor and include a list of attribute(s)
#(Shape_Length, NAME, POP2005, REGION) in the parameter(s)
```

```
rows = arcpy.da.SearchCursor(inputdata, ["Shape_Length", "NAME", "POP2005",
"REGION"])

# iterate through the rows in the cursor
# attributes are accessed using rows[index]
# rows[0] is "Shape_Length"
# rows[1] is NAME
# rows[2] is POP2005
for row in rows:
      print(row)
```

SearchCursor using ***with*** statement ensures the close of iterator:

```
inputdata = "H:\\Lab9\\EPI.gdb\\world_country_poly"
with arcpy.da.SearchCursor(inputdata, ["Shape_Length", "NAME", "POP2005",
"REGION"]) as rows
for row in rows:
      print (row)
```

Next is an example of how to update a field using cursors. Note that you did this in the Python lab (earlier in the semester in the fourth assignment) in a different way.

Note that this code shows you how to use an update cursor but it's not very meaningful! The shape length attribute is going to be converted quite arbitrarily. The code updates the field "Shape_Length" under the conditions of the field "UN" being greater than 100.

```
inputdata = "H:\\Lab9\\EPI.gdb\\world_country_poly"
# Add a new field PopCat
arcpy.management.AddField("world_country_poly","PopCat","FLOAT")
# Then calculate its value based on a different field POP2005
with arcpy.da.UpdateCursor(inputdata, ["PopCat", "POP2005"]) as rows:
      for row in rows:
            print(row)
            # update the field "PopCat" based on Population POP2005 being greater
than 25 Million or not
            if row[1] > 25000000:
                  row[0] = 1
            else:
                  row[0] = 2
            rows.updateRow(row)
            #print(row)
```

You can also add new data as shown below. Here, you insert a cursor and list the attributes that need to be filled with new values:

```
#Inserting new data
inputdata = " H:\\Lab9\\students.shp"

cursor = arcpy.da.InsertCursor(inputdata, ["ID", "SHAPE@XY"])
new_row = ["40", (-71.128, 42.376)]
cursor.insertRow(new_row)
```

Once you have tried out all the code, think about how you would use cursors to calculate population density (POP2005/AREA) for each country in the world country shapefile? Remember that you should only calculate population density where area is greater than 0. (You don't want to divide by zero).

Hint: Modify the update cursor. You should add a new density field using `arcpy.management.AddFields`

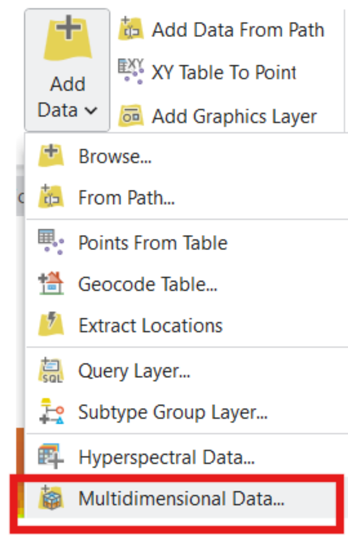
Part 3 Question (1 point)

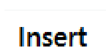
Attach a python script that uses cursors and a screenshot of the table associated with world countries showing the newly calculated population density. (1 point)

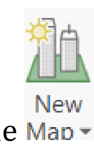
4.0 Netcdf data and Python

The netcdf data you will use in this part of the exercise was downloaded from this website: **http://www.ncdc.noaa.gov/** . I selected global precipitation from the `Global Precipitation Climatology Project (GPCP)` **aggregated data** where I selected the time range from 1979 to 2014[4]. It is documented here: **http://eagle1.umd.edu/GPCP_ICDR/GPCP_Background.html**

ArcGIS's help notes that "NetCDF (network Common Data Form)[5] is a file format for storing multidimensional scientific data (variables) such as temperature, humidity, pressure, wind speed, and direction. Each of these variables can be displayed through a dimension (such as time) in ArcGIS by making a layer or table view from the netCDF file". Note that if you had temperature that varied by height from sea level you could use height as a dimension.

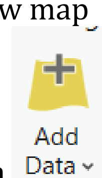


27. In the **Insert** 

pane 

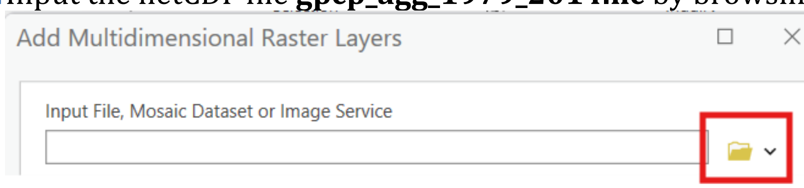
insert a new map

28. In the **Map** 

pane, click on **Add Data** 

and select the multidimensional data layer.

29. Input the netCDF file **gpcp_agg_1979_2014.nc** by browsing to it

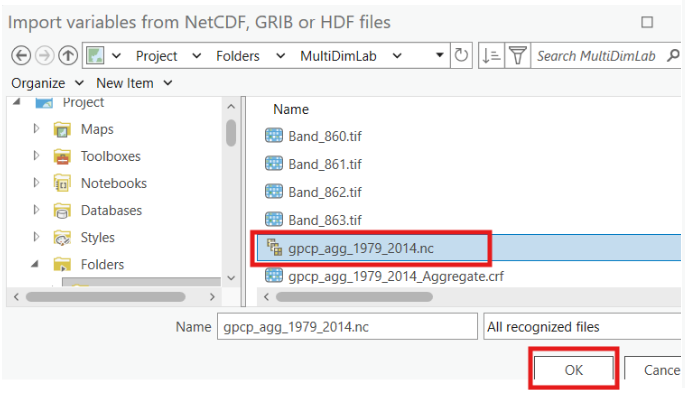


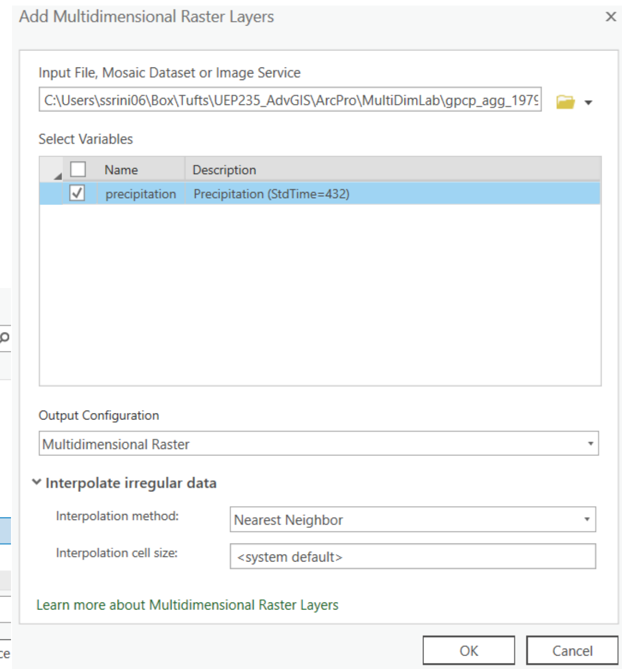
. Note that it will not be visible in the gdb but it is in the folder where you have extracted the data for this exercise.

---

[4] A more recent version is here: https://www.ncei.noaa.gov/thredds/catalog/cdr/gpcp_final_agg/catalog.html?dataset=cdr/gpcp_final_agg/CGPC_Final_Aggregation_best.ncd.

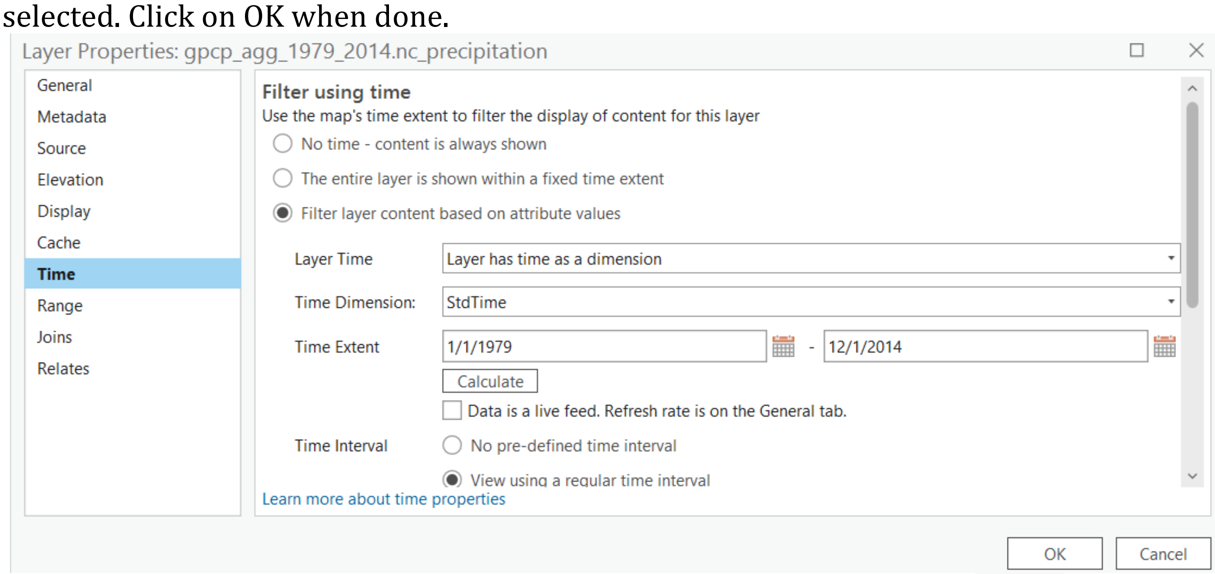
[5] **http://www.unidata.ucar.edu/software/netcdf/**

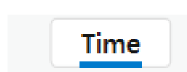


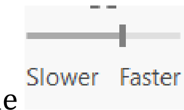


The layer should automatically be added to the display.

30. Then right click on this newly added layer in the **Contents** and select properties to get a new window.
31. In the **Time** tab **check** change the no time to “Layer has time as a dimension” if it does not happen automatically. Select **Stdtime** as the “Time dimension” if it hasn’t already been selected. Click on OK when done.

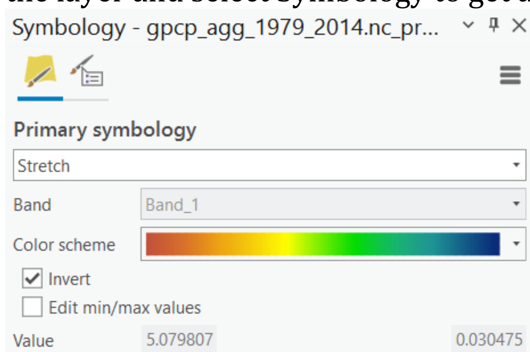


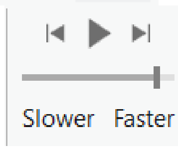
32. In the Time tab on the main menu ribbon  make sure that  time is enabled

33. Change the speed of the animation using the 

controls for it

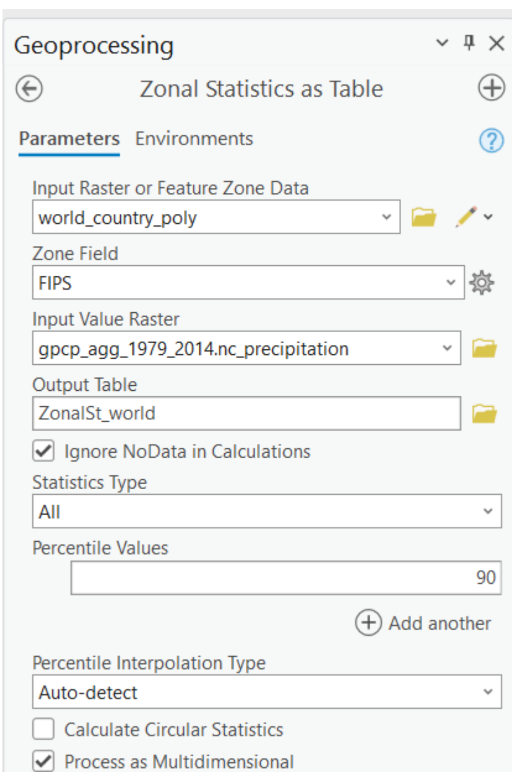
34. You can also reset the symbology with a standard palette for precipitation. Right click on the layer and select Symbology to get a window for it. Then change the color scheme

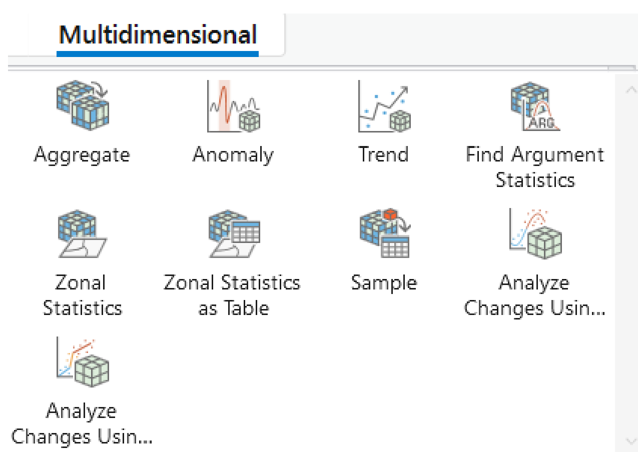


35. You can click on the button 

to see how precipitation changes.

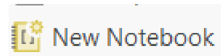
The easiest way to record this animation is to use Snagit in the Data Lab. However, you can also record it using the Animation tools in ArcGIS as described in the previous tutorial.

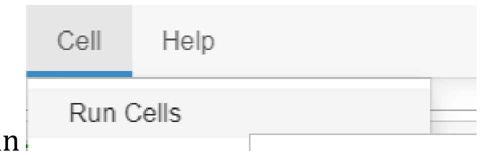


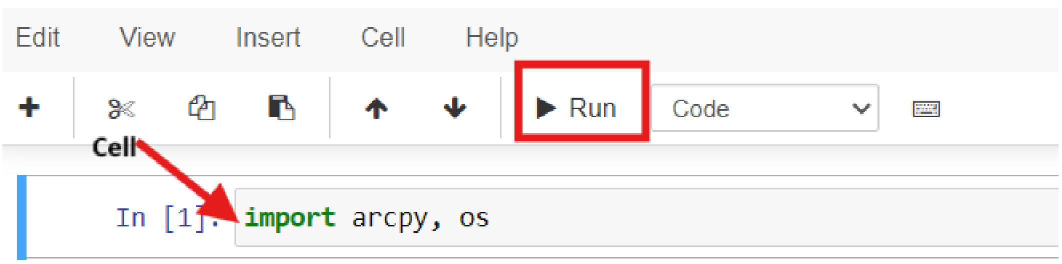


In the Multidimensional pane you can calculate an array of statistics including zonal statistics as table which will calculate for every country, the average precipitation over time. In the code

for this (saved as precipitation.py) you will reproduce a tool like this in Python. Note that the data in the netcdf are roughly two readings every month over the 36-year time-period from 1979-2014. Assume that country boundaries remain the same. First try to understand what the code will do and then try the notebook in ArcGIS Pro. See how to insert your own notebook here:
https://pro.arcgis.com/en/pro-app/arcpy/get-started/pro-notebooks.htm

36. Go to the Insert menu and then select New Notebook 

to insert a notebook
37. You can copy and paste the code that is attached into this line by line as shown below or simply open the notebook that is already saved in the data folder.
38. To run the code go to the cell, then click on Run 



39. If it does not work you will get an error message:

```
---------------------------------------------------------------------------
RuntimeError                              Traceback (most recent call last)
Cell In[4], line 1
----> 1 rows = arcpy.da.SearchCursor(inputdata, ["Shape_Length", "NAME", "POP2005"])
      2 for row in rows:
      3         print(row)

RuntimeError: cannot open 'C:\Users\ssrini06\Box\Tufts\UEP235_AdvGIS\ArcPro\MultiDimLab\EPI.gdb\world_country_p
```

40. It it works it will not have any error messages in the Out as shown below

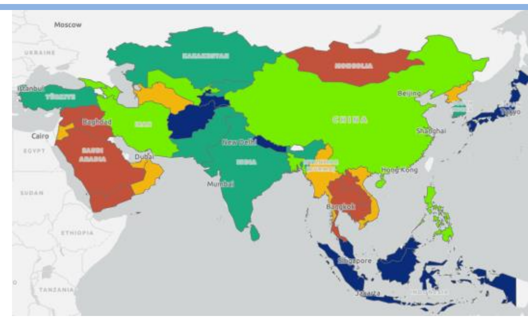
```
[18]: # Execute MakeNetCDFRasterLayer
      arcpy.MakeNetCDFRasterLayer_md(Input_NetCDF,variable, XDimension, YDimension,
                                     OutRasterLayer, bandDimension, dimensionValues,
                                     valueSelectionMethod)
```

[18]: Messages

Start Time: Monday, November 3, 2025 2:14:21 PM

Succeeded at Monday, November 3, 2025 2:14:23 PM (Elapsed Time: 1.70 seconds)

Part 4 Question (2 points)



Attach a screenshot of your final table(s) and map showing precipitation in Asian countries as well as your code in Python where you modified the code to calculate the average precipitation for countries in Asia for a different set of intervals in 1979-2014 than the last 10. Explain how, for example, you could get every fourth year from 1979 onwards or every March from 1979 onwards. (2 points)