Creating a Python Toolbox for Script Customization in ArcGIS Pro

Catlin Corrales

3200K Programming for Geospatial Science & Technology

Dr. Huidae Cho

August 6, 2021

Abstract

Using Geographic Information Systems (GIS) offers local governments an avenue by which to store, analyze, and visualize geospatial data. ESRI's ArcGIS software is becoming an increasingly common way to do this, as it offers both desktop and online options. However, geoprocessing can be time consuming and repetitive when working with city or county data. Consequently, an option to automate and customize geoprocessing workflows is ideal. Using a Python Toolbox allows users to automate both the analysis and mapping of geospatial projects and ultimately results in increased efficiency. Proposed in this project is a Python Toolbox called Population Density Calculator that can potentially be used to calculate population density at the block group level using U.S. Census Bureau tabular and geographic data from The National Historical Geographic Information System (NHGIS).

1. Introduction

GIS is important in the field of urban and regional planning because it can facilitate analyses that can be used to help inform planners where or where not to start development projects (Somers, 1987). One such analysis is determining population density on census tract, census block or census block group level. Conducting this type of demographic analysis on a smaller scale is often more effective than conducting it on the city or county scale (Cohen, Hatchard and Wilson, 2015). For bigger geographic regions, population density values are less likely to provide accurate measures due to modifiable areal unit problems (MAUP) that occur in GIS analysis (Cohen, Hatchard and Wilson, 2015). Therefore, the aim of this project is to create a theoretical population density calculator on the census block group level that can be used to analyze neighborhood growth and population trends.

A Python Toolbox can be used to customize and automate this process. By creating a script for a Python Toolbox, not only does the user make the geoprocessing workflow more efficient, but they can also share the Python Toolbox .pyt file with planners in other regions (Van Rees, 2014, 46-47). The concept of creating tools that can be shared, replicated, or improved upon is good practice to contribute to the overall betterment of programing.

2. Materials and Methods

To conduct this project, a subscription to ArcGIS was required. In this case, the software used was ArcGIS Pro. As this was an exploratory project, a test batch of census tabular and geographic data for a southern California city were acquired from the NHGIS, and a geographic boundary of an AOI of the city of Yucaipa was obtained from the San Bernardino County GIS website (cms.sbcounty.gov, n.d.).

2.1 Workflow

The geoprocessing workflow to conduct this analysis consisted of the following steps: using the Add Join tool to append the census tabular and geographic data, using the Clip tool to extract our AOI from the state dataset, using the Add Field tool to add a population density field to the attribute table, and using the Calculate field tool to run an expression to populate the newly added field. The population density per census block group was determined by dividing the total population of each block group by the shape area of each bock group polygon.

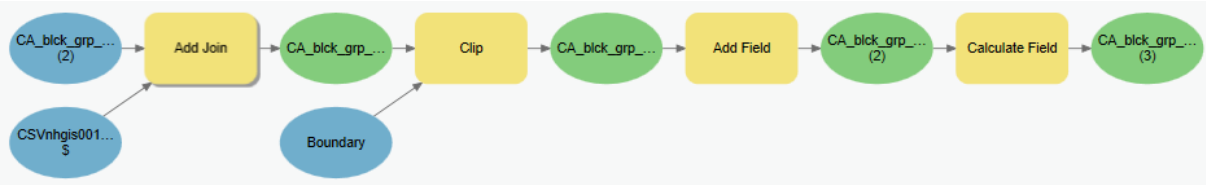

Figure 1. Model for the workflow used in the Python Toolbox

2.2 Python Script

The ESRI Python Toolbox template was used as a base to create the Population Density Calculator (pro.arcgis.com, n.d. *The Python toolbox template—ArcGIS Pro*). The script was then written using ArcPy, an ESRI Python site package (pro.arcgis.com, n.d. *What is ArcPy?—ArcGIS Pro*). Once complete, the .pyt file was then connected to the ArcPro project catalog pane Toolboxes folder for use. The following is the pseudocode of the script used to create the Population Density Python Toolbox. It includes the ArcPy syntax that is required to execute the script in ArcGIS.

```
START
import arcpy
#Define the toolbox and name of .pyt file
class Toolbox(object):
   Function (self):
      self.label = name of toolbox
      # List of tool classes associated with this toolbox
      self.tools = name of all tools in toolbox
#Define the tool and name of the class
class Name of tool number 1(object):
   Function __init__(self):
      self.label = name of tool
      self.description = ""
      self.canRunInBackground = False
   #Define all parameter definitions
   Function getParameterInfo(self):
      param1 = arcpy.Parameter(
        describe param)
      param2 = arcpy.Parameter(
         describe param)
      param3 = arcpy.Parameter(
         describe param)
      param4 = arcpy.Parameter(
          describe param)
      params = list all params
      return params
   Function isLicensed(self):
      return True
   Function updateParameters(self, parameters):
      return
```

```
    #source code of the tool
    Function execute(self, parameters, messages):
        param1 = parameters[0].valueAsText
        param2 = parameters[1].valueAsText
        param3 = parameters[2].valueAsText
        param4 = parameters[3].valueAsText
        joined = arcpy.management.AddJoin(param1, join field, param2, join field)
        arcpy.analysis.Clip(joined, param3, param4)
        arcpy.management.AddField(param4, field name, data type)
        arcpy.management.CalculateField(param4, field name, expression)
        return
    END
```

Figure 2. Pseudocode script used for the Population Density Python Toolbox

3. Results

The resulting Python Toolbox Population Density Calculator was able to successfully execute all of the required functions.

3.1 User Interface

When added to the ArcPro project, the Python Toolbox Population Density Calculator created a user-friendly interface that was clearly labeled and simple to use.

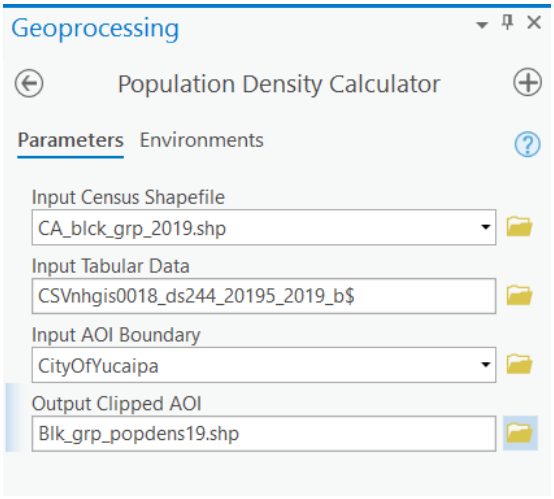

Figure 3. The ArcGIS Pro user interface that resulting from the Python Toolbox

4. Discussion

The customization and automation of the theoretical geoprocessing workflow was successfully completed by using an ArcGIS Python Toolbox. The resulting user interface was easy to use and has the potential to increase efficiency by eliminating repetitive tasks that would otherwise be necessary to run the workflow.

4.1 Future Steps

While the Python Toolbox was able to run successfully, it can be further improved to make the workflow even more seamless. The resulting layer produced by the Population Density Calculator creates an image of the AOI containing all of the relevant data in its attribute table. However, in order to create a visually interpretable image, the user must still manually change the symbology of the AOI to show the PopDens field in a graduated color choropleth map (Figure 4). Further script customization is required to incorporate these symbology changes into the Python Toolbox.

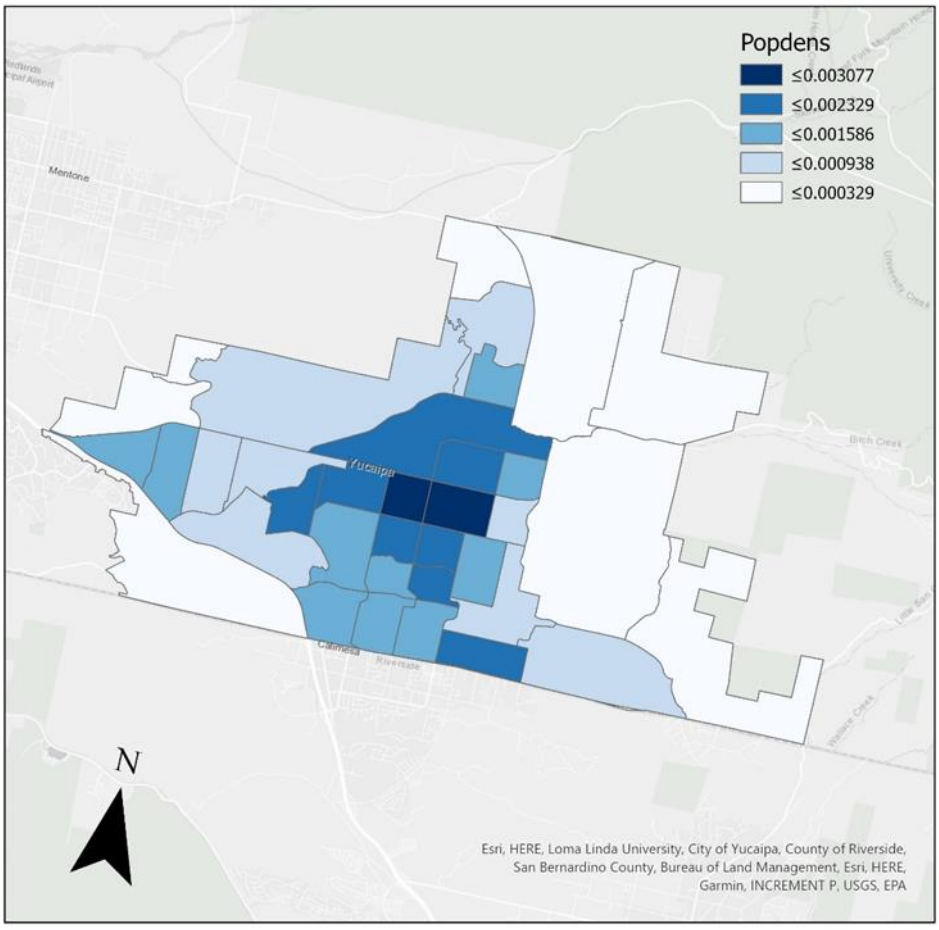

Figure 4. An image of the final output once the symbology changes are incorporated into the existing Population Density Calculator .pyt script.

5. Conclusion

The overall creation, customization and execution of the Python Toolbox proved to be successful in automating a theoretical workflow and increasing efficiency for running demographic analyses. While the script for this toolbox would benefit from further customization, it is important to note that the implementation of a Python Toolbox has the potential to be used as a powerful tool to aid in urban and regional planning. There are vast different categories of geoprocessing tools available in the ArcPy module that can be tailored to fit the needs of other types of analyses as well.

References

cms.sbcounty.gov. (n.d.). *Geographic Information Systems (GIS) > Home*. [online] Available at: http://cms.sbcounty.gov/gis/Home.aspx [Accessed 1 Aug. 2021].

Cohen, D.T., Hatchard, G.W. and Wilson, S.G. (2015). *Population Trends in Incorporated Places: 2000 to 2013 Population Estimates and Projections Current Population Reports*. [online] Available at: https://www.census.gov/content/dam/Census/library/publications/2015/demo/p25-1142.pdf [Accessed 4 Aug. 2021].

desktop.arcgis.com. (n.d.). *What is a Python toolbox?—Help | ArcGIS for Desktop*. [online] Available at: https://desktop.arcgis.com/en/arcmap/10.3/analyze/creating-tools/a-quick-tour-of-python-toolboxes.htm [Accessed 4 Aug. 2021].

pro.arcgis.com. (n.d.). *Add Field (Data Management)—ArcGIS Pro | Documentation*. [online] Available at: https://pro.arcgis.com/en/pro-app/latest/tool-reference/data-management/add-field.htm [Accessed 3 Aug. 2021].

pro.arcgis.com. (n.d.). *Add Join (Data Management)—ArcGIS Pro | Documentation*. [online] Available at: https://pro.arcgis.com/en/pro-app/latest/tool-reference/data-management/add-join.htm [Accessed 4 Aug. 2021].

pro.arcgis.com. (n.d.). *Calculate Field (Data Management)—ArcGIS Pro | Documentation*. [online] Available at: https://pro.arcgis.com/en/pro-app/latest/tool-reference/data-management/calculate-field.htm [Accessed 4 Aug. 2021].

pro.arcgis.com. (n.d.). *Clip (Analysis)—ArcGIS Pro | Documentation*. [online] Available at: https://pro.arcgis.com/en/pro-app/latest/tool-reference/analysis/clip.htm [Accessed 3 Aug. 2021].

pro.arcgis.com. (n.d.). *Defining parameter data types in a Python toolbox—ArcGIS Pro | Documentation*. [online] Available at: https://pro.arcgis.com/en/pro-app/latest/arcpy/geoprocessing_and_python/defining-parameter-data-types-in-a-python-toolbox.htm [Accessed 3 Aug. 2021].

pro.arcgis.com. (n.d.). *Parameter—ArcGIS Pro | Documentation*. [online] Available at: https://pro.arcgis.com/en/pro-app/latest/arcpy/classes/parameter.htm# [Accessed 3 Aug. 2021].

pro.arcgis.com. (n.d.). *The Python toolbox template—ArcGIS Pro | Documentation*. [online] Available at: https://pro.arcgis.com/en/pro-app/latest/arcpy/geoprocessing_and_python/a-template-for-python-toolboxes.htm.

pro.arcgis.com. (n.d.). *What is ArcPy?—ArcGIS Pro | Documentation*. [online] Available at: https://pro.arcgis.com/en/pro-app/latest/arcpy/get-started/what-is-arcpy-.htm [Accessed 4 Aug. 2021].

Somers, R. (1987). *Geographic Information Systems in Local Government: A Commentary for Photogrammetry and Remote Sensing Multipurpose Cadastre, Simplified FIG. 1. Local Government Data Models*. [online]. Available at: https://www.asprs.org/wp-content/uploads/pers/1987journal/oct/1987_oct_1379-1382.pdf [Accessed 4 Aug. 2021].

Steven Manson, Jonathan Schroeder, David Van Riper, Tracy Kugler, and Steven Ruggles. IPUMS National Historical Geographic Information System: Version 16.0 [dataset]. Minneapolis, MN: IPUMS. 2021. http://doi.org/10.18128/D050.V16.0

Toms, S. and O'beirne, D. (2017). *ArcPy and ArcGIS : automating ArcGIS for desktop and ArcGIS online with Python*. [online] Birmingham: Packt Publishing. Available at: https://www.perlego.com/book/527192/arcpy-and-arcgis-second-edition-pdf.

Van Rees, E. 2014, "Python Scripting and GIS Increasing Efficiency", *GeoInformatics*, vol. 17, no. 7, pp. 46-47.