Python Script for Esri API

The previous *Python* script was written for generate the OD matrix through *Google Map API* in 2014. In this project, I rewrote the *Python* code to generate an updated OD matrix using *Esri API* instead, named “CalcDis.py”. The workflow is as follow.

Once the table is prepared, import the table in the database to your local csv files. Then, one can use the *Esri API* for *Python* and the *Pandas* data frame to generate the reduced station network. In the python script, the input csv file must include a lat and a lon field in the “test1.csv” file, and it will output a file named “test2.csv” that has the calculated the distance and travel time between all pairs of points. I did not set up any specific parameter in the code, so it will output a result using the current traffic to calculate the travel time as it is in the default setting.

Per the *Esri* documentation, the use limits for a regular user is 1,000 point and 3600 seconds. I successfully generated 10 pairs in 21.2 seconds, 100 pairs in 66.5 seconds, and 1,000 pairs in 728.2 seconds. However, it seems that I reach the limits of a regular user very soon, then I have to use the *SAML* login as an ASU student user who has 1,000 assigned credits. The *Python* script examples can be found via this link:

<https://developers.arcgis.com/python/guide/part5-generate-od-cost-matrix/>

*“Try: except”* syntax helps me with the error handling because sometimes Esri API couldn’t find a route between the two points and the program stops executing. The syntax skipped the points where there is no route found between them and output a null value. However, I use a print function there to show the index so I can keep track of where the errors happen.

The energy-saving mode should be turned off, so it will not sleep every few minutes, which pauses my program before it finishes. One should avoid using the mode so the program will not be interrupted.

The *Jupyter Notebook* is a dynamic tool for scratching and testing your *Python* code. I use *Anaconda* to load the *Jupyter Notebook*, which avoids many issues when launching it in the Mac os *Terminal*.