Tutorial

Computational environment: Python 3.8.5

Third-party environment: pandas, geopandas, numpy, matplotlib, scipy

This project mainly contains 2 files: ECM.py and my\_package.py.

ECM.py is the main script file to realize the function of calculating human current, resistance, and so on. Moreover, my\_package.py encapsulated the necessary tools to finish the data analysis.

Detailed introduction for script file:

**ECM.py**

1. preprocess\_data():

Input arguments: {input\_file\_path: file path of CSV of **one day** GPS data fulfill the format of Figure 1 (one file correspond to one day data), output\_file\_folder: file path of CSV of preprocessed result}

Output: A series of files (each time period correspond to each file) of population and summation of velocity of direction and direction of each node (CSV file)

Content: This function contains the necessary process of data preprocessing, including removing outliers, transforming hour and minute to second, transforming unit of velocity from m/s to km/h, decomposing velocity to x direction and y direction, converting longitude and latitude to relative coordinate to find out which user belongs to which node (this part code won’t work directly for other countries than Japan, modification is required), cutting a day into many time period per 30 minutes, calculating population and summation of velocity x and y at each node, merging result into one result file.

Note: Name of corresponding variable vary from people to people, modification is required.

1. cal\_current():

Input arguments: {input\_file\_path: file path of CSV of preprocessed data calculated by preprocess\_data()}

Output: Human current (pandas dataframe format)

Content: calculate human current from each node.

Note: population \* mean velocity is equivalent to the summation of velocity, therefore we used later in our code instead.

1. cal\_CTS ():

Input argument: {weekday\_list: calendar of which day’s want to calculate}

Output: time series of human current

Content: Merge result calculated by cal\_current() together to form a time series dataframe.

1. cal\_resistance():

Input argument: {month\_list: which date data want to use to inference human resistance}

Output: Human resistance (pandas dataframe format)

Content: Use formula proposed in the paper (the inverse of mean value of absolute value of human current) to calculate human resistance for each node.

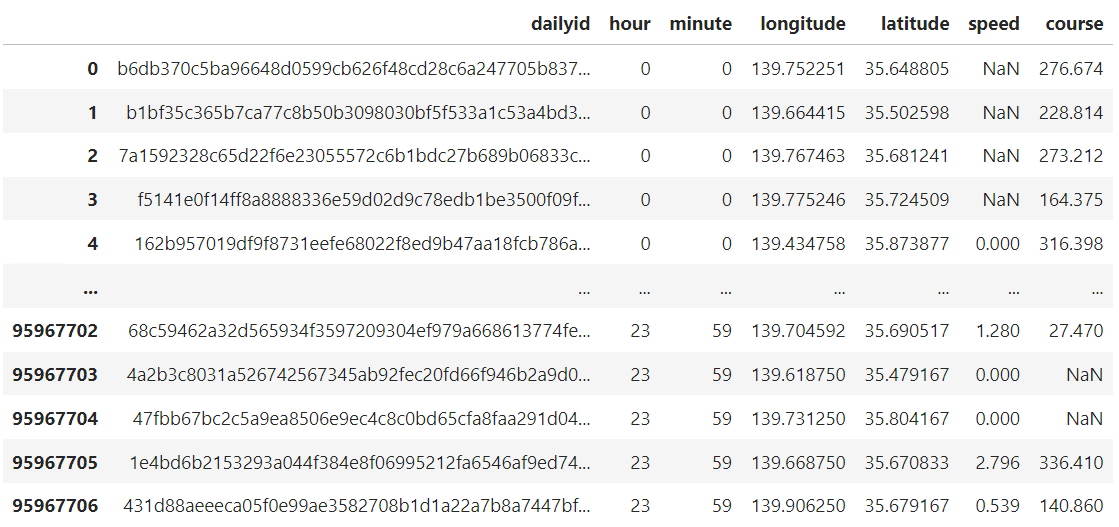


Figure 1. example of raw GPS data (Note: speed and course are not necessary for raw data because it can be calculated if (user, time, longitude, latitude) are known)

**my\_package.py**

1. longitude\_to\_id\_x():

Input argument: {longitude: longitude}

Output: relative coordinate x

Content: map longitude to Cartesian coordinate system x (cut a map into many small meshes)

Note: Coordinate origin is (lon: 122.93, lat= 20.42), the far west south territory of Japan. Other counties case need to modify before use.

1. latitude\_to\_id\_y():

Input argument: { latitude: latitude }

Output: relative coordinate y

Content: map longitude to Cartesian coordinate system y (cut a map into many small meshes)

Note: Coordinate origin is (lon: 122.93, lat= 20.42), the far west south territory of Japan. Other counties case need to modify before use.

1. id\_x\_to\_longitude(id\_x):

Input argument: { id\_x: relative coordinate x}

Output: longitude

Content: get original longitude from relative coordinate x

Note: When trying to plot the node on a map, can use this method to get the original longitude of the node.

1. id\_y\_to\_latitude(id\_y):

Input argument: { id\_y: relative coordinate y}

Output: latitude

Content: get original latitude from relative coordinate y

Note: When trying to plot the node on a map, can use this method to get the original latitude of the node.

1. right\_node(), left\_node(), up\_node(), down\_node():

Input argument: { current\_id: current relative coordinate (format: ‘x,y’)}

Output: coordinate (ID) of right (left, up, and down) node of current node

Content: Can find neighbour node of give node.