In [63]:
```
#import packages
import pandas as pd
import statsmodels.api as sm
import plotly.express as plt
import geopandas as gpd
import plotly.graph_objects as go
import matplotlib.pyplot as mp
from shapely.geometry import Point
import numpy as np
```

In [237]:
```
#read in data
cities = pd.read_csv('pset3_cities.csv')
stations = pd.read_csv('pset3_stations.csv')
lines = pd.read_csv('pset3_lines.csv')
```

1.a

In [238]:
```
#merge lines and stations
stations2 = stations.merge(lines, how = 'left')
stations = stations.merge(lines, how = 'left')
stations['year_opening'] = stations['year_opening'].replace([2017, 201
stations = stations.dropna()
stations['year_opening'].describe()
```

Out[238]:
```
count     339.000000
mean     2012.846608
std         2.179805
min      2008.000000
25%      2011.000000
50%      2013.000000
75%      2015.000000
max      2016.000000
Name: year_opening, dtype: float64
```

In [239]:
```
#get iteration tools
stationsdum = pd.get_dummies(stations['cityid'])
statdum = pd.get_dummies(stations2['cityid'])
```

In [240]:
```
#create gik, Si, qik

#initiating variables
stationsdum.head()
cities['numstat'] = cities['cityid']
cities['numlinks'] = cities['cityid']
cities['avgspd'] = cities['cityid']
for i in [1, 2, 3, 4, 5, 6, 7, 8, 9, 18]:
    cities['x_' + str(i)] = cities['cityid'].astype(str) +'_'+ str(i)
a = []

#looping over cities
for i in cities['cityid']:
    if i in stationsdum.columns:
        #filling values
        cities['numstat'] = cities['numstat'].replace(i, stationsdum[i
        cities['numlinks'] = cities['numlinks'].replace(i, stations[st
        cities['avgspd'] = cities['avgspd'].replace(i, stations[statio
        citiestr = cities[cities['cityid'] == i]
        for h in [1, 2, 3, 4, 5, 6, 7, 8, 9, 18]:
            cities['x_' + str(h)] = cities['x_' + str(h)].replace(str(

    elif i in statdum.columns:
        #filling values
        cities['numstat'] = cities['numstat'].replace(i, 0)
        cities['numlinks'] = cities['numlinks'].replace(i, 0)
        cities['avgspd'] = cities['avgspd'].replace(i, None)
        for h in [1, 2, 3, 4, 5, 6, 7, 8, 9, 18]:
            cities['x_' + str(h)] = cities['x_' + str(h)].replace(str(

    else:
        #filling empties
        cities['numstat'] = cities['numstat'].replace(i, 0)
        cities['numlinks'] = cities['numlinks'].replace(i, 0)
        cities['avgspd'] = cities['avgspd'].replace(i, None)
        for h in [1, 2, 3, 4, 5, 6, 7, 8, 9, 18]:
            cities['x_' + str(h)] = cities['x_' + str(h)].replace(str(
```

In [249]:
```
#A

#summarizing delta_lines
print(cities['numstat'].describe())
```

```
count    340.000000
mean       0.997059
std        1.143143
min        0.000000
25%        0.000000
50%        1.000000
75%        1.250000
max        7.000000
Name: numstat, dtype: float64
```

1.b

In [232]:
```
#getting region dummies
regions = pd.get_dummies(cities['province_en'])
cities = pd.concat([cities, regions], axis = 1)
```

In [233]:
```
# B with fixed effects

citiesempt = cities.dropna(subset = ['empgrowth'])

treat = ['numstat']
dummies = list(set(list(cities['province_en'])))
vals = sm.OLS(citiesempt['empgrowth'], citiesempt[treat + dummies])
out = vals.fit(cov_type = 'HC0')
print(out.summary())
```

```
                            OLS Regression Results
======================================================================
=========
Dep. Variable:              empgrowth   R-squared:
0.480
Model:                            OLS   Adj. R-squared:
0.416
Method:                 Least Squares   F-statistic:
nan
Date:                Sat, 18 Nov 2023   Prob (F-statistic):
nan
Time:                        19:49:45   Log-Likelihood:
74.782
No. Observations:                 275   AIC:
-87.56
Df Residuals:                     244   BIC:
24.55
Df Model:                          30
Covariance Type:                  HC0
======================================================================
=============
                    coef    std err          z      P>|z|      [0.02
5      0.975]
----------------------------------------------------------------------
-------------
numstat           0.0496      0.014      3.576      0.000       0.02
2       0.077
qinghai           0.1558      0.007     22.480      0.000       0.14
2       0.169
qinghai           0.1558      0.007     22.480      0.000       0.14
2       0.169
shandong          0.1245      0.019      6.452      0.000       0.08
7       0.162
shandong          0.1245      0.019      6.452      0.000       0.08
7       0.162
guangdong         0.1518      0.024      6.442      0.000       0.10
6       0.198
guangdong         0.1518      0.024      6.442      0.000       0.10
6       0.198
liaoning          0.0140      0.021      0.678      0.498      -0.02
6       0.054
liaoning          0.0140      0.021      0.678      0.498      -0.02
6       0.054
guangxi           0.0905      0.021      4.390      0.000       0.05
0       0.131
guangxi           0.0905      0.021      4.390      0.000       0.05
```

```
0        0.131
yunnan              0.1064       0.022      4.918      0.000       0.06
4        0.149
yunnan              0.1064       0.022      4.918      0.000       0.06
4        0.149
xinjiang            0.1418       0.058      2.440      0.015       0.02
8        0.256
xinjiang            0.1418       0.058      2.440      0.015       0.02
8        0.256
anhui               0.1407       0.027      5.150      0.000       0.08
7        0.194
anhui               0.1407       0.027      5.150      0.000       0.08
7        0.194
beijing             0.1432       0.021      6.887      0.000       0.10
2        0.184
beijing             0.1432       0.021      6.887      0.000       0.10
2        0.184
fujian              0.0993       0.033      3.003      0.003       0.03
5        0.164
fujian              0.0993       0.033      3.003      0.003       0.03
5        0.164
zhejiang            0.1359       0.037      3.635      0.000       0.06
3        0.209
zhejiang            0.1359       0.037      3.635      0.000       0.06
3        0.209
hunan               0.0420       0.028      1.526      0.127      -0.01
2        0.096
hunan               0.0420       0.028      1.526      0.127      -0.01
2        0.096
hebei               0.0644       0.022      2.878      0.004       0.02
1        0.108
hebei               0.0644       0.022      2.878      0.004       0.02
1        0.108
hubei               0.2431       0.044      5.578      0.000       0.15
8        0.329
hubei               0.2431       0.044      5.578      0.000       0.15
8        0.329
heilongjiang       -0.1482       0.031     -4.836      0.000      -0.20
8       -0.088
heilongjiang       -0.1482       0.031     -4.836      0.000      -0.20
8       -0.088
chongqing           0.1757       0.028      6.339      0.000       0.12
1        0.230
chongqing           0.1757       0.028      6.339      0.000       0.12
1        0.230
tibet               0.2052    5.65e-16   3.63e+14      0.000       0.20
5        0.205
tibet               0.2052    5.65e-16   3.63e+14      0.000       0.20
5        0.205
jiangsu             0.2217       0.032      6.972      0.000       0.15
9        0.284
jiangsu             0.2217       0.032      6.972      0.000       0.15
9        0.284
guizhou             0.1209       0.026      4.563      0.000       0.06
9        0.173
guizhou             0.1209       0.026      4.563      0.000       0.06
```

```
9       0.173
inner mongolia     0.0549      0.038      1.455      0.146      -0.01
9       0.129
inner mongolia     0.0549      0.038      1.455      0.146      -0.01
9       0.129
shanxi             0.0483      0.023      2.144      0.032       0.00
4       0.092
shanxi             0.0483      0.023      2.144      0.032       0.00
4       0.092
ningxia            0.0508      0.016      3.268      0.001       0.02
0       0.081
ningxia            0.0508      0.016      3.268      0.001       0.02
0       0.081
shanghai           0.2799      0.021     13.465      0.000       0.23
9       0.321
shanghai           0.2799      0.021     13.465      0.000       0.23
9       0.321
jilin              0.0599      0.025      2.406      0.016       0.01
1       0.109
jilin              0.0599      0.025      2.406      0.016       0.01
1       0.109
jiangxi            0.1870      0.027      6.980      0.000       0.13
5       0.240
jiangxi            0.1870      0.027      6.980      0.000       0.13
5       0.240
gansu              0.1251      0.019      6.479      0.000       0.08
7       0.163
gansu              0.1251      0.019      6.479      0.000       0.08
7       0.163
sichuan            0.1018      0.034      3.003      0.003       0.03
5       0.168
sichuan            0.1018      0.034      3.003      0.003       0.03
5       0.168
shaanxi            0.1320      0.026      5.081      0.000       0.08
1       0.183
shaanxi            0.1320      0.026      5.081      0.000       0.08
1       0.183
henan              0.1713      0.019      8.843      0.000       0.13
3       0.209
henan              0.1713      0.019      8.843      0.000       0.13
3       0.209
tianjin            0.0804      0.028      2.900      0.004       0.02
6       0.135
tianjin            0.0804      0.028      2.900      0.004       0.02
6       0.135
======================================================================
=========
Omnibus:                       12.517   Durbin-Watson:
2.157
Prob(Omnibus):                  0.002   Jarque-Bera (JB):
19.555
Skew:                           0.291   Prob(JB):
5.67e-05
Kurtosis:                       4.170   Cond. No.
2.74e+16
======================================================================
```

```
=========

Notes:
[1] Standard Errors are heteroscedasticity robust (HC0)
[2] The smallest eigenvalue is 9.79e-31. This might indicate that the
re are
strong multicollinearity problems or that the design matrix is singul
ar.
```

In [234]:
```
#B no FE

citiesempt = cities.dropna(subset = ['empgrowth'])

treat = ['numstat']

vals = sm.OLS(citiesempt['empgrowth'], sm.add_constant(citiesempt[trea
out = vals.fit(cov_type = 'HC0')
print(out.summary())
```

```
                            OLS Regression Results
==============================================================================
=========
Dep. Variable:              empgrowth   R-squared:
0.123
Model:                            OLS   Adj. R-squared:
0.120
Method:                 Least Squares   F-statistic:
35.15
Date:                Sat, 18 Nov 2023   Prob (F-statistic):
9.19e-09
Time:                        19:49:45   Log-Likelihood:
2.9709
No. Observations:                 275   AIC:
-1.942
Df Residuals:                     273   BIC:
5.292
Df Model:                           1
Covariance Type:                  HC0
==============================================================================
=========
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
---------
const          0.1812      0.021      8.529      0.000       0.140
0.223
numstat        0.0765      0.013      5.929      0.000       0.051
0.102
==============================================================================
=========
Omnibus:                        7.984   Durbin-Watson:
1.452
Prob(Omnibus):                  0.018   Jarque-Bera (JB):
13.741
Skew:                          -0.053   Prob(JB):
0.00104
Kurtosis:                       4.090   Cond. No.
2.71
==============================================================================
=========

Notes:
[1] Standard Errors are heteroscedasticity robust (HC0)
```

1.d

In [274]:
```
#C/D

citiesempt = cities.dropna(subset = ['empgrowth'])


treat = ['numstat']
controls = ['x_1', 'x_2', 'x_3', 'x_4', 'x_5', 'x_6', 'x_7', 'x_8','x_
vals = sm.OLS(citiesempt['empgrowth'], sm.add_constant(citiesempt[trea
out = vals.fit(cov_type = 'HC0')
print(out.summary())

cities['pred'] = out.predict(sm.add_constant(cities[treat + controls])
cities['resid'] = cities['pred'] - cities['empgrowth']
```

```
                            OLS Regression Results
==============================================================================
=========
Dep. Variable:              empgrowth   R-squared:
0.202
Model:                            OLS   Adj. R-squared:
0.169
Method:                 Least Squares   F-statistic:
7.683
Date:                Sun, 19 Nov 2023   Prob (F-statistic):
1.69e-11
Time:                        16:03:51   Log-Likelihood:
15.905
No. Observations:                 275   AIC:
-7.810
Df Residuals:                     263   BIC:
35.59
Df Model:                          11
Covariance Type:                  HC0
==============================================================================
=========
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
---------
const          0.1625      0.024      6.665      0.000       0.115
0.210
numstat        0.0210      0.022      0.951      0.342      -0.022
0.064
x_1           -0.0006      0.023     -0.028      0.978      -0.045
0.044
x_2            0.0077      0.029      0.263      0.792      -0.050
0.065
x_3            0.0286      0.029      0.985      0.324      -0.028
0.086
x_4            0.0105      0.034      0.312      0.755      -0.056
0.077
x_5            0.0520      0.041      1.278      0.201      -0.028
0.132
x_6            0.1089      0.029      3.712      0.000       0.051
0.166
x_7            0.2145      0.046      4.697      0.000       0.125
```

```
0.304
x_8            0.1418      0.072      1.977      0.048       0.001
0.282
x_9            0.0611      0.043      1.414      0.157      -0.024
0.146
x_18           0.0865      0.046      1.875      0.061      -0.004
0.177
======================================================================
=========
Omnibus:                        9.348   Durbin-Watson:
1.534
Prob(Omnibus):                  0.009   Jarque-Bera (JB):
17.638
Skew:                          -0.054   Prob(JB):
0.000148
Kurtosis:                       4.236   Cond. No.
11.2
======================================================================
=========

Notes:
[1] Standard Errors are heteroscedasticity robust (HC0)
```

1.e

In [288]:
```
#predict opening on line speed
vals = sm.WLS(lines['open']/lines['open'].std(), sm.add_constant(lines
out = vals.fit(cov_type = 'HC0')
print(out.summary())
```

```
                            WLS Regression Results
==============================================================================
=========
Dep. Variable:                   open   R-squared:
0.001
Model:                            WLS   Adj. R-squared:
-0.006
Method:                 Least Squares   F-statistic:
0.1067
Date:                Sun, 19 Nov 2023   Prob (F-statistic):
0.744
Time:                        16:48:07   Log-Likelihood:
-227.19
No. Observations:                 149   AIC:
458.4
Df Residuals:                     147   BIC:
464.4
Df Model:                           1
Covariance Type:                  HC0
==============================================================================
=========
                 coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
---------
const          1.0884      0.474      2.294      0.022       0.159
2.018
speed          0.0006      0.002      0.327      0.744      -0.003
0.004
==============================================================================
=========
Omnibus:                      109.788   Durbin-Watson:
0.628
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
12.851
Skew:                          -0.307   Prob(JB):
0.00162
Kurtosis:                       1.699   Cond. No.
1.45e+03
==============================================================================
=========

Notes:
[1] Standard Errors are heteroscedasticity robust (HC0)
[2] The condition number is large, 1.45e+03. This might indicate that
there are
strong multicollinearity or other numerical problems.
```

In [289]:
```
#predict number of open stations on distance to beijing
vals = sm.OLS(cities['numstat']/cities['numstat'].std(), sm.add_consta
out = vals.fit(cov_type = 'HC0')
print(out.summary())
```

```
                            OLS Regression Results
==============================================================================
=========
Dep. Variable:                numstat   R-squared:
0.005
Model:                            OLS   Adj. R-squared:
0.003
Method:                 Least Squares   F-statistic:
2.374
Date:                Sun, 19 Nov 2023   Prob (F-statistic):
0.124
Time:                        16:48:09   Log-Likelihood:
-481.00
No. Observations:                 340   AIC:
966.0
Df Residuals:                     338   BIC:
973.7
Df Model:                           1
Covariance Type:                  HC0
==============================================================================
===========
                  coef    std err          z      P>|z|      [0.025
0.975]
------------------------------------------------------------------------------
-----------
const           1.0136      0.109      9.273      0.000       0.799
1.228
dist_beijing   -0.0001   7.31e-05     -1.541      0.123      -0.000
3.06e-05
==============================================================================
=========
Omnibus:                      113.436   Durbin-Watson:
1.451
Prob(Omnibus):                  0.000   Jarque-Bera (JB):
308.031
Skew:                           1.577   Prob(JB):
1.29e-67
Kurtosis:                       6.435   Cond. No.
3.05e+03
==============================================================================
=========

Notes:
[1] Standard Errors are heteroscedasticity robust (HC0)
[2] The condition number is large, 3.05e+03. This might indicate that
there are
strong multicollinearity or other numerical problems.
```

1.f

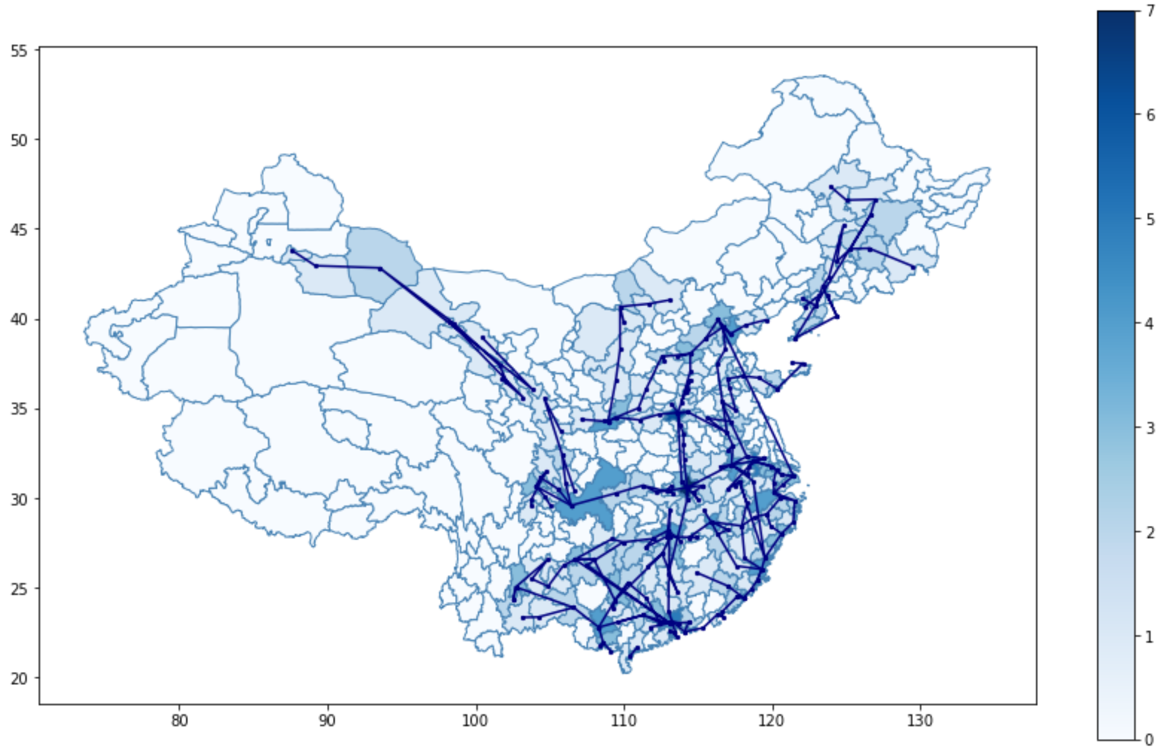
In [276]:
```
#map nonsense

#open shapefile
shapefile = gpd.read_file("chn_admbnda_adm2_ocha.shp")
```

In [277]:
```
#merge cities and stations
merge1 = stations.merge(lines)

cities2 = cities.merge(stations, on = 'cityid')
```

In [281]:
```
#turn merge into geofile
cities2['numlinks'] = cities2['numlinks'].replace(0, np.nan)

cities2 = cities2.dropna(subset = ['numlinks'])

geometry = [Point(xy) for xy in zip(cities['longitude'], cities['latit
gdf1 = gpd.GeoDataFrame(cities, crs=crs, geometry=geometry)

#join shapefile and new geofile
shapefile2 = gpd.sjoin(gdf1, shapefile, how="right", op='intersects')

#print base colored by number of open stations
base = shapefile2.plot(figsize=(15,9), column = 'numstat', edgecolor='

geometry = [Point(xy) for xy in zip(cities2['longitude'], cities2['lat

crs = {'init': shapefile1.crs}

gdf = gpd.GeoDataFrame(cities2, crs=crs, geometry=geometry)
gdf.plot(ax= base, marker = 'o', markersize = 5, color = 'navy')

#plotting open lines
for i in list(set(list(lines['lineid']))):
    trunc = cities2[cities2['lineid']==i]
    mp.plot(trunc.longitude, trunc.latitude, color = 'navy')
```

```
/opt/anaconda3/lib/python3.8/site-packages/pyproj/crs/crs.py:141: Fut
ureWarning:

'+init=<authority>:<code>' syntax is deprecated. '<authority>:<code>'
is the preferred initialization method. When making the change, be mi
ndful of axis order changes: https://pyproj4.github.io/pyproj/stable/
gotchas.html#axis-order-changes-in-proj-6 (https://pyproj4.github.io/
pyproj/stable/gotchas.html#axis-order-changes-in-proj-6)

/opt/anaconda3/lib/python3.8/site-packages/IPython/core/interactivesh
ell.py:3357: FutureWarning:

The `op` parameter is deprecated and will be removed in a future rele
ase. Please use the `predicate` parameter instead.

<ipython-input-281-edbb9c49cc6a>:10: UserWarning:

CRS mismatch between the CRS of left geometries and the CRS of right
geometries.
Use `to_crs()` to reproject one of the input geometries to match the
CRS of the other.

Left CRS: +init=epsg:4326 +type=crs
Right CRS: EPSG:4326


/opt/anaconda3/lib/python3.8/site-packages/pyproj/crs/crs.py:141: Fut
ureWarning:

'+init=<authority>:<code>' syntax is deprecated. '<authority>:<code>'
```

is the preferred initialization method. When making the change, be mindful of axis order changes: https://pyproj4.github.io/pyproj/stable/gotchas.html#axis-order-changes-in-proj-6 (https://pyproj4.github.io/pyproj/stable/gotchas.html#axis-order-changes-in-proj-6)



In [282]:
```
#turn merge into geofile
cities2['numlinks'] = cities2['numlinks'].replace(0, np.nan)

cities2['resid'].fillna(-5)

cities2 = cities2.dropna(subset = ['numlinks'])

geometry = [Point(xy) for xy in zip(cities['longitude'], cities['latit
gdf1 = gpd.GeoDataFrame(cities, crs=crs, geometry=geometry)

#join shapefile and new geofile
shapefile2 = gpd.sjoin(gdf1, shapefile, how="right", op='intersects')

#print base colored by number of open stations
base = shapefile2.plot(figsize=(15,9), column = 'resid', edgecolor='st

geometry = [Point(xy) for xy in zip(cities2['longitude'], cities2['lat

crs = {'init': shapefile1.crs}

gdf = gpd.GeoDataFrame(cities2, crs=crs, geometry=geometry)
gdf.plot(ax= base, marker = 'o', markersize = 5, color = 'navy')

#plotting open lines
for i in list(set(list(lines['lineid']))):
    trunc = cities2[cities2['lineid']==i]
    mp.plot(trunc.longitude, trunc.latitude, color = 'navy')
```

```
/opt/anaconda3/lib/python3.8/site-packages/pyproj/crs/crs.py:141: Fut
ureWarning:

'+init=<authority>:<code>' syntax is deprecated. '<authority>:<code>'
is the preferred initialization method. When making the change, be mi
ndful of axis order changes: https://pyproj4.github.io/pyproj/stable/
gotchas.html#axis-order-changes-in-proj-6 (https://pyproj4.github.io/
pyproj/stable/gotchas.html#axis-order-changes-in-proj-6)

/opt/anaconda3/lib/python3.8/site-packages/IPython/core/interactivesh
ell.py:3357: FutureWarning:

The `op` parameter is deprecated and will be removed in a future rele
ase. Please use the `predicate` parameter instead.

<ipython-input-282-6dcd3b901323>:12: UserWarning:

CRS mismatch between the CRS of left geometries and the CRS of right
geometries.
```

In [ ]: