02_explore

February 4, 2019

In [1]:
```
from IPython.display import HTML
```

1 *Explore* spatial data

In [2]:
```
%matplotlib inline

        import pandas as pd
        import geopandas as gpd
        from pysal.lib import weights
        from pysal.explore import esda
        from pysal.viz.splot.esda import plot_moran, plot_local_autocorrelation, lisa_cluster
        import contextily as ctx
        import matplotlib.pyplot as plt

        db = gpd.read_file('../data/demo_data.gpkg')
```

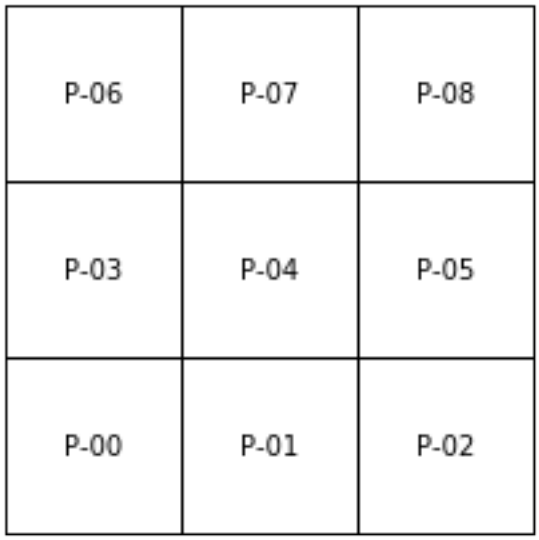
```
/opt/conda/lib/python3.6/site-packages/pysal/model/spvcm/abstracts.py:10: UserWarning: The `dill
  from .sqlite import head_to_sql, start_sql
```

1.1 Spatial weights

1.1.1 Lattice example

In [3]:
```
import numpy as np
        from shapely.geometry import Polygon

        # do a regular 3x3 lattice and draw it here
        w = weights.lat2W(3, 3, rook=True)
        # Get points in a grid
        l = np.arange(3)
        xs, ys = np.meshgrid(l, l)
        # Set up store
        polys = []
        # Generate polygons
        for x, y in zip(xs.flatten(), ys.flatten()):
            poly = Polygon([(x, y), (x+1, y), (x+1, y+1), (x, y+1)])
```

```
    polys.append(poly)
# Convert to GeoSeries
polys = gpd.GeoSeries(polys)
gdf = gpd.GeoDataFrame({'geometry': polys,
                        'id': ['P-%s'%str(i).zfill(2) for i in range(len(polys))]})
w.remap_ids(gdf['id'].values)
# Annotate & Visualise
ax = polys.plot(edgecolor='k', facecolor='w')
[plt.text(x, y, t,
          verticalalignment='center',
          horizontalalignment='center') for x, y, t in zip(
         [p.centroid.x for p in polys],
         [p.centroid.y for p in polys],
         [i for i in gdf['id']])]
ax.set_axis_off()
```



- Full matrix

```
In [4]: pd.DataFrame(w.full()[0],
                     index=gdf['id'],
                     columns=gdf['id'],
                    ).astype(int)
```

Out[4]:

id	P-00	P-01	P-02	P-03	P-04	P-05	P-06	P-07	P-08
id									
P-00	0	1	0	1	0	0	0	0	0

P-01	1	0	1	0	1	0	0	0	0
P-02	0	1	0	0	0	1	0	0	0
P-03	1	0	0	0	1	0	1	0	0
P-04	0	1	0	1	0	1	0	1	0
P-05	0	0	1	0	1	0	0	0	1
P-06	0	0	0	1	0	0	0	1	0
P-07	0	0	0	0	1	0	1	0	1
P-08	0	0	0	0	0	1	0	1	0

1.1.2 Real-word data

```
In [5]: w_queen = weights.Queen.from_dataframe(db)
```

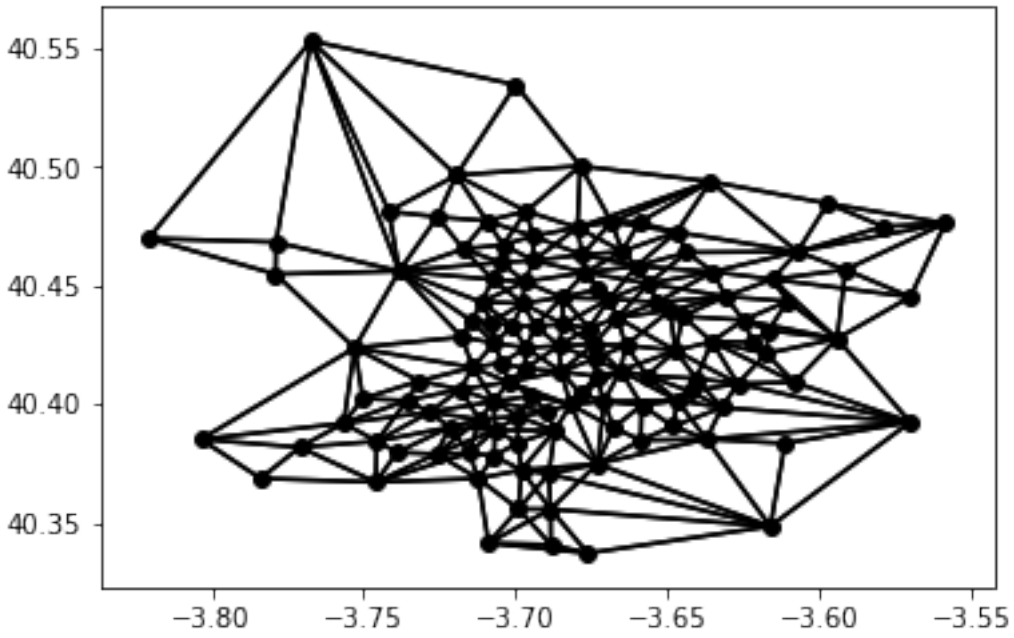
```
In [6]: w_k1 = weights.KNN.from_dataframe(db, k=1)
```

```
/opt/conda/lib/python3.6/site-packages/pysal/lib/weights/weights.py:170: UserWarning: The weight
  warnings.warn("The weights matrix is not fully connected. There are %d components" % self.n_co
```

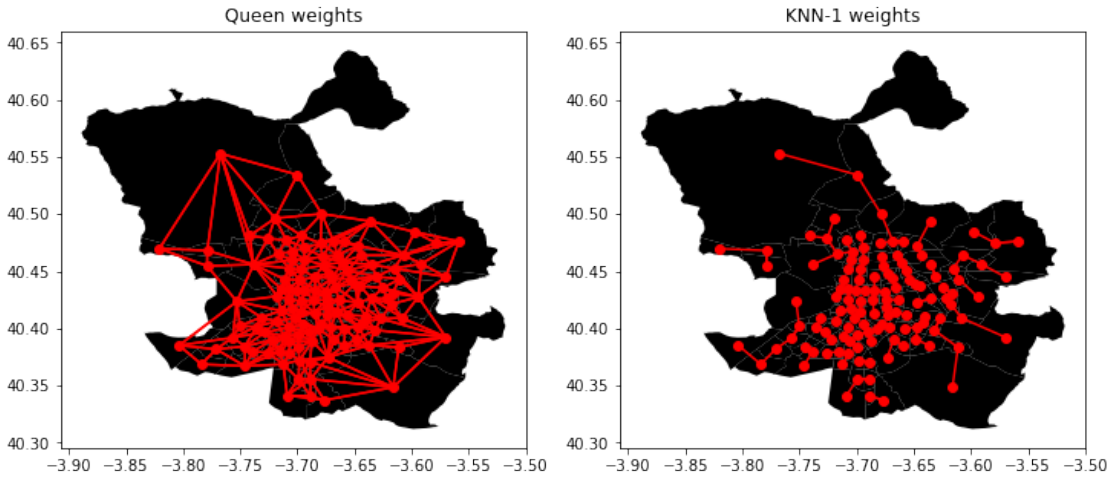
- Visualising weights

```
In [7]: w_queen.plot(db)
```

```
Out[7]: (<Figure size 432x288 with 1 Axes>,
         <matplotlib.axes._subplots.AxesSubplot at 0x7fa3146b01d0>)
```



```
In [8]: f, axs = plt.subplots(1, 2, figsize=(12, 6))

        db.plot(color='k', ax=axs[0])
        w_queen.plot(db, ax=axs[0], color='red')
        axs[0].set_title('Queen weights')

        db.plot(color='k', ax=axs[1])
        w_k1.plot(db, ax=axs[1], color='red')
        axs[1].set_title('KNN-1 weights')

        plt.show()
```



- And fancier graphs are also possible…

```
In [9]: blob = """
                <blockquote class="twitter-tweet" data-lang="en">
                <p lang="en" dir="ltr">
                    A few more mesmerizing bundled graphs from Spatial Weights Matrices and the
                    <a href="https://twitter.com/hashtag/PySAL?src=hash&amp;ref_src=twsrc%5Etfw">
                    #PySAL</a>/<a href="https://twitter.com/datashader?ref_src=twsrc%5Etfw">@data
                    code I used. Tbh, I&#39;m not sure bundles are more useful at identifying the
                    spatial structure than traditional graphs, what do people think?
                    <a href="https://t.co/UkaPZlvUqL">https://t.co/UkaPZlvUqL</a>
                    <a href="https://t.co/3QW7zOIaCB">pic.twitter.com/3QW7zOIaCB</a></p>&mdash;
                    Dani Arribas-Bel (@darribas)
                    <a href="https://twitter.com/darribas/status/1056293955422294016?ref_src=twsr
                    October 27, 2018</a></blockquote>
                 <script async src="https://platform.twitter.com/widgets.js"
                 charset="utf-8"></script>
                """
        HTML(blob)
```

```
Out[9]: <IPython.core.display.HTML object>
```

CHALLENGE - Create a spatial weights matrix using the KNN algorithm and picking the average number of neighbors under the queen criterium (hint: checkout the `mean_neighbors` attribute in `w_queen`)

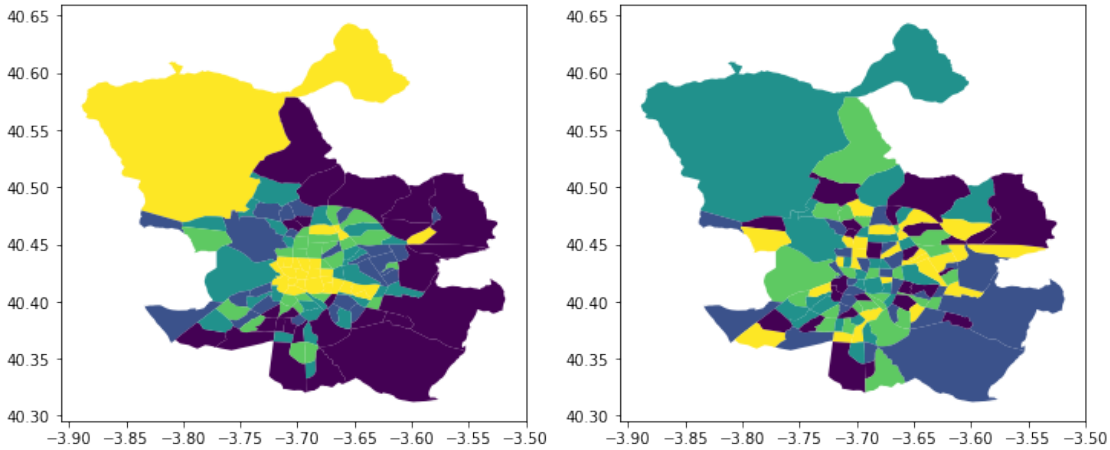
1.2 Global Spatial autocorrelation

- Shuffle values across space

```
In [10]: np.random.seed(1234)
         db['arturo_score_shuffled'] = db['arturo_score'].sample(frac=1).values
```

- Same values, different geographical implications...

```
In [11]: f, axs = plt.subplots(1, 2, figsize=(12, 6))
         db.plot(column='arturo_score', scheme='quantiles', ax=axs[0])
         db.plot(column='arturo_score_shuffled', scheme='quantiles', ax=axs[1])
         plt.show()
```



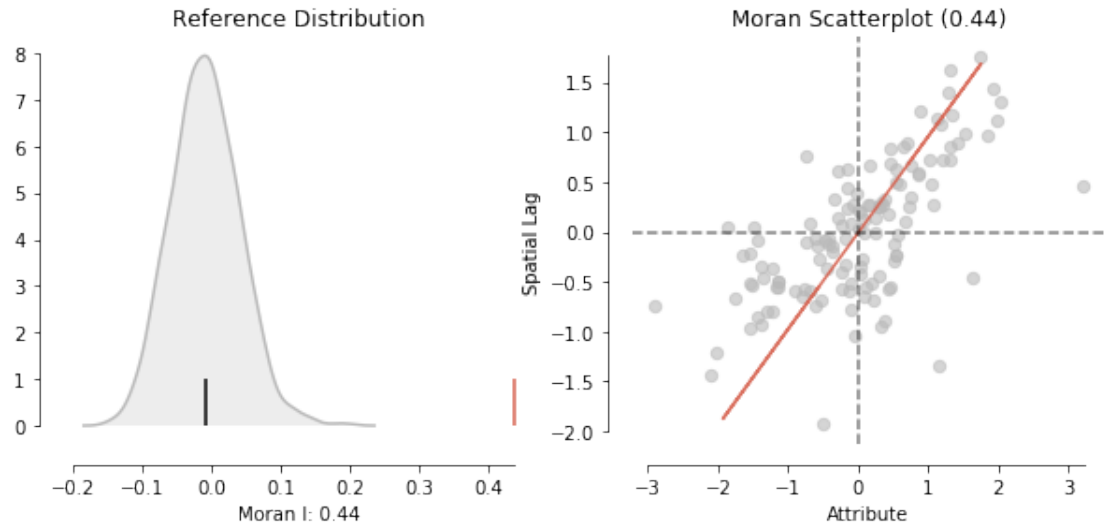
- Moran's I

```
In [12]: moran = esda.Moran(db['arturo_score'], w_queen)

In [13]: moran.I

Out[13]: 0.43867621491355296

In [14]: moran.p_sim

Out[14]: 0.001
```

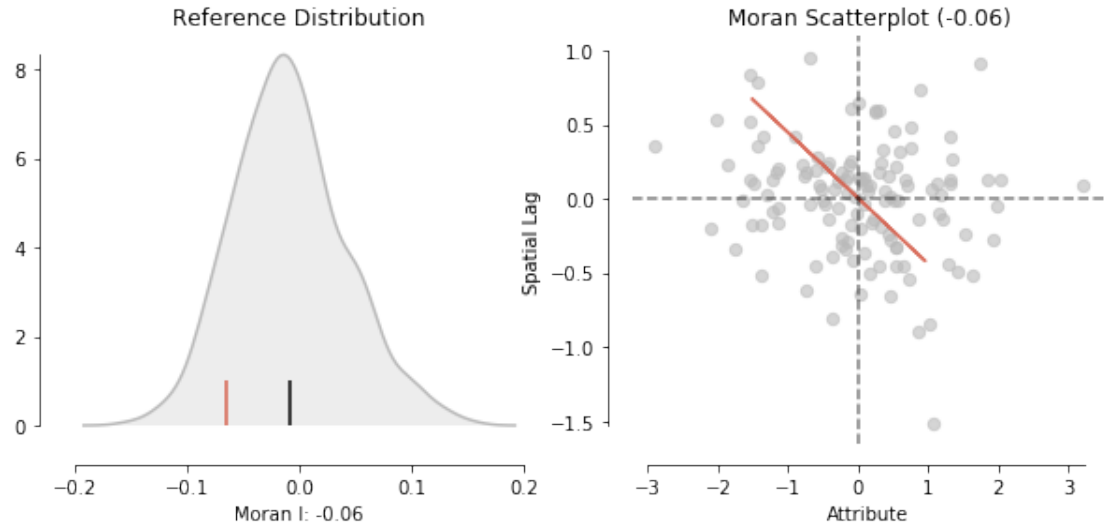
```
In [15]: moran_shuffled = esda.Moran(db['arturo_score_shuffled'], w_queen)

In [16]: moran_shuffled.I

Out[16]: -0.06423418028115105

In [17]: moran_shuffled.p_sim

Out[17]: 0.124
```

- Moran Scatterplots

```
In [18]: plot_moran(moran);
```




```
In [19]: plot_moran(moran_shuffled);
```



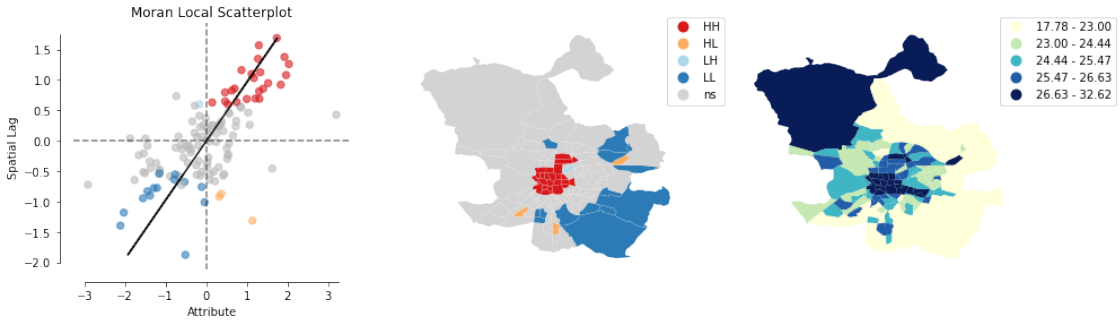
CHALLENGE - Compute Moran's I for the average price in AirBnb (`abb_price_usd`) and explore its scatterplot

1.3 Local Spatial autocorrelation

```
In [20]: lisa = esda.Moran_Local(db['arturo_score'], w_queen)
```

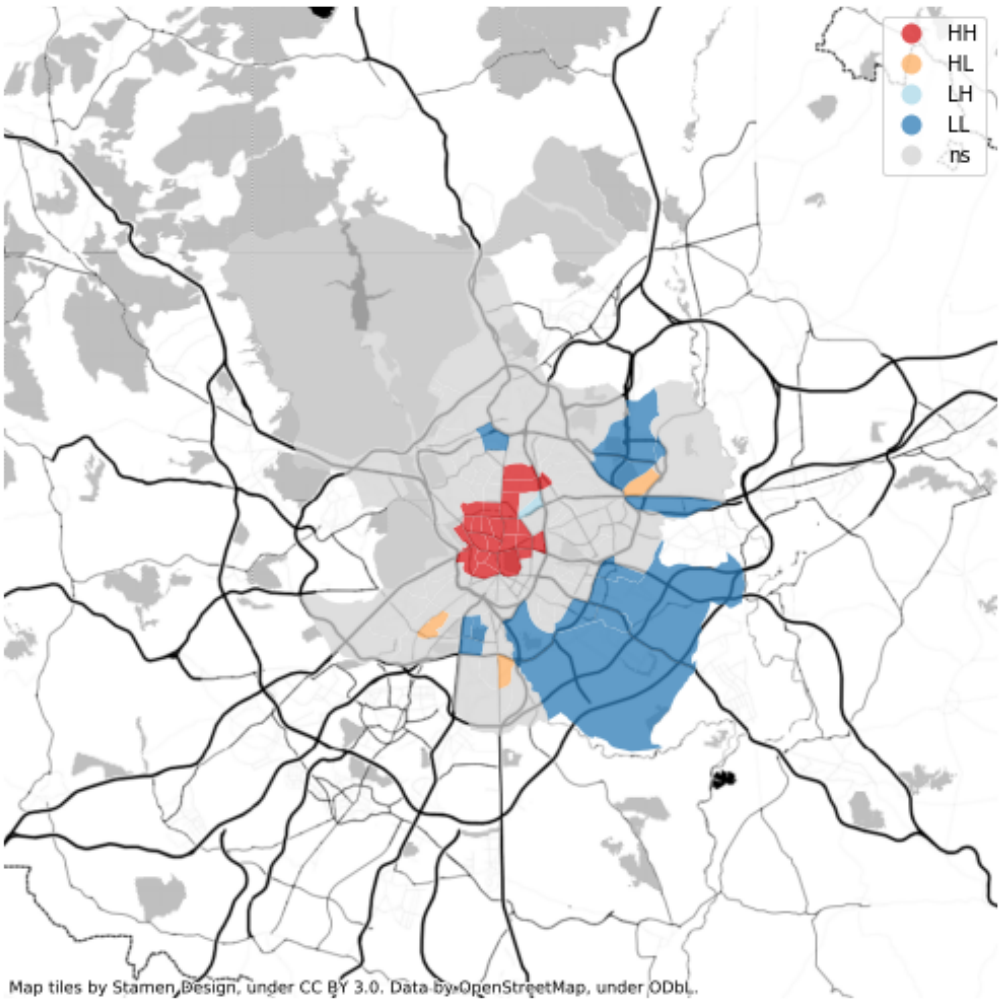
- Overall plot

```
In [21]: plot_local_autocorrelation(lisa, db, 'arturo_score');
```



- Cluster map

```
In [22]: f, ax = plt.subplots(1, figsize=(9, 9))
         lisa_cluster(lisa, db.to_crs(epsg=3857), alpha=0.75, ax=ax)
         ctx.add_basemap(ax, url=ctx.sources.ST_TONER_BACKGROUND)
         plt.show()
```



CHALLENGE - Create a similar LISA map for population density (`population_density`)