> Geographic Data Science

with

PySAL

and the

pydata stack

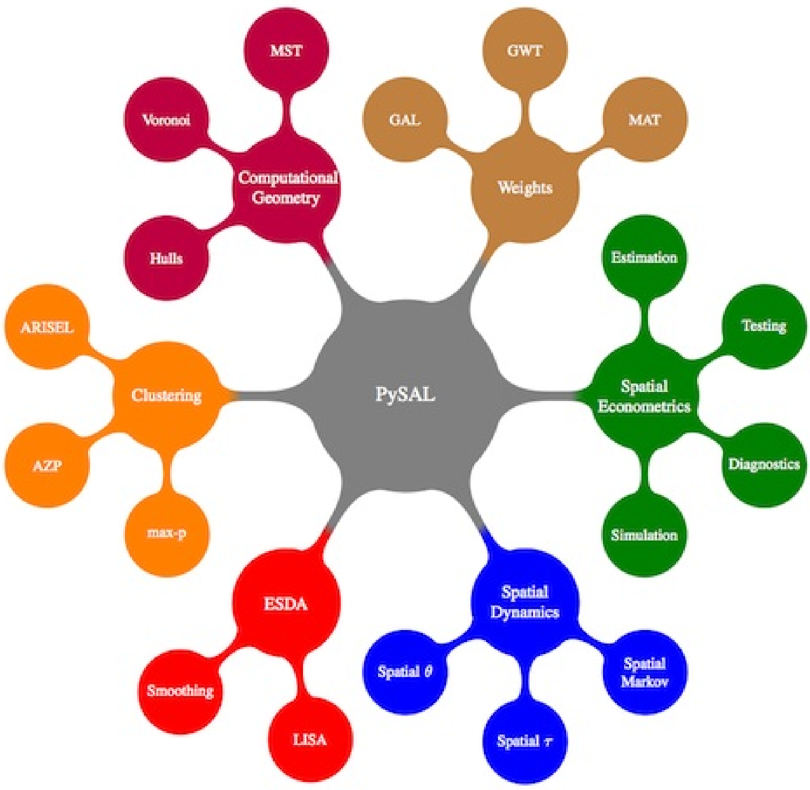
Sergio J. Rey

Dani Arribas-Bel

Table of Contents

Geographic Data Science with PySAL and the pydata stack

This two-part tutorial will first provide participants with a gentle introduction to Python for geospatial analysis, and an introduction to version `PySAL` 1.11 and the related eco-system of libraries to facilitate common tasks for Geographic Data Scientists. The first part will cover munging geo-data and exploring relations over space. This includes importing data in different formats (e.g. shapefile, GeoJSON), visualizing, combining and tidying them up for analysis, and will use libraries such as `pandas` , `geopandas` , `PySAL` , or `rasterio` . The second part will provide a gentle overview to demonstrate several techniques that allow to extract geospatial insight from the data. This includes spatial clustering and regression and point pattern analysis, and will use libraries such as `PySAL` , `scikit-learn` , or `clusterpy` . A particular emphasis will be set on presenting concepts through visualization, for which libraries such as `matplotlib` , `seaborn` , and `folium` will be used.



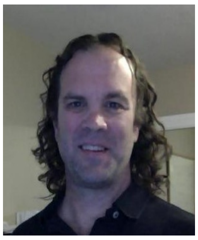
Geographic Data Science Lab

Distribution

> [URL] [PDF] [EPUB] [MOBI] [IPYNB]

License

About the authors

 Sergio Rey is professor of geographical sciences and core faculty member of the GeoDa Center for Geospatial Analysis and Computation at the Arizona State University. His research interests include open science, spatial and spatio-temporal data analysis, spatial econometrics, visualization, high performance geocomputation, spatial inequality dynamics, integrated multiregional modeling, and regional science. He co-founded the Python Spatial Analysis Library (PySAL) in 2007 and continues to direct the PySAL project. Rey is a fellow of the spatial econometrics association and editor of the journal Geographical Analysis.

 Dani Arribas-Bel is Lecturer in Geographic Data Science and member of the Geographic Data Science Lab at the University of Liverpool (UK). Dani is interested in undestanding cities as well as in the quantitative and computational methods required to leverage the power of the large amount of urban data increasingly becoming available. He is also part of the team of core developers of PySAL, the open-source library written in Python for spatial analysis. Dani regularly teaches Geographic Data Science and Python courses at the University of Liverpool and has designed and developed several workshops at different levels on spatial analysis and econometrics, Python and open source scientific computing.

Outline

Part I

1. Software and Tools Installation (10 min)
2. Spatial data processing with PySAL (45 min)

   a. Input-output

   b. Visualization and Mapping

   c. Spatial weights
3. Exercise (10 min.)
4. ESDA with PySAL (45 min)

   a. Global Autocorrelation

   b. Local Autocorrelation

   c. Space-Time exploratory analysis
5. Exercise (10 min)

Part II

1. Point Patterns (30 min)

   a. Kernel Density Estimation visualization

   b. Centrography and distance based statistics
2. Exercise (10 min)
3. Spatial clustering a (30 min)

   a. Geodemographic analysis

   b. Regionalization
4. Exercise (30 min)
5. Spatial Regression (30 min)

   a. Overview

   c. Basic spatial regression: spatial lag and error model
6. Exercise (10 min)

Data

This tutorial makes use of a variety of data sources. Below is a brief description of each dataset as well as the links to the original source where the data was downloaded from. For convenience, we have repackaged the data and included them in the compressed file with the notebooks. You can download it here.

AirBnb listing for Austin (TX)

Source: Inside AirBnb's extract of AirBnb locations in Austin (TX).

Part I

Software and Tools Installation

Dependencies

Participants should have installed the following dependencies:

- Anaconda or MiniConda Python distributions for Python 2.7. See installation instructions on the links.
- `git`
- A `conda` environment loaded with all the dependencies can be installed by running the `pydata.sh` script available as part of the `envs` repository (Github link). To install it, follow these instructions:
    - Clone the repository on your machine:

      `> git clone https://github.com/darribas/envs.git`

    - Navigate into the folder:

      `> cd envs`

    - Run the script:

      `> bash pydata.sh`

Once installed, you need to activate the environment to run the notebooks. In Windows, open up PowerShell and type:

`> activate pydata`

And if you are on GNU/Linux or OSX:

`> source activate pydata`

Get started

Instructions to fire up a notebook here.

Spatial Data Processing

Notebook here.

```
%matplotlib inline

import pysal as ps
```

ESDA with `PySAL`

```
%matplotlib inline

import pysal as ps
```

```
/home/dani/anaconda/envs/pydata/lib/python2.7/site-packages/matplotlib/font_manager.py:
273: UserWarning: Matplotlib is building the font cache using fc-list. This may take a
moment.
  warnings.warn('Matplotlib is building the font cache using fc-list. This may take a m
oment.')
```

Part II

Point Patterns

> IPYNB

This notebook covers a brief introduction on how to visualize and analyze point patterns. To demonstrate this, we will use a dataset of all the AirBnb listings in the city of Austin (check the Data section for more information about the dataset).

Before anything, let us load up the libraries we will use:

```
%matplotlib inline

import numpy as np
import pandas as pd
import geopandas as gpd
import seaborn as sns
import matplotlib.pyplot as plt
import mplleaflet as mpll
```

Data preparation

Let us first set the paths to the datasets we will be using:

```
# Adjust this to point to the right file in your computer
listings_link = '/home/dani/Desktop/listings.csv'
```

The core dataset we will use is `listings.csv`, which contains a lot of information about each individual location listed at AirBnb within Austin:

```
lst = pd.read_csv(listings_link)
lst.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5835 entries, 0 to 5834
Data columns (total 92 columns):
id                                    5835 non-null int64
listing_url                           5835 non-null object
scrape_id                             5835 non-null int64
last_scraped                          5835 non-null object
name                                  5835 non-null object
summary                               5373 non-null object
space                                 4475 non-null object
description                           5832 non-null object
experiences_offered                   5835 non-null object
neighborhood_overview                 3572 non-null object
notes                                 2413 non-null object
transit                               3492 non-null object
```

```
thumbnail_url                     5542 non-null object
medium_url                        5542 non-null object
picture_url                       5835 non-null object
xl_picture_url                    5542 non-null object
host_id                           5835 non-null int64
host_url                          5835 non-null object
host_name                         5820 non-null object
host_since                        5820 non-null object
host_location                     5810 non-null object
host_about                        3975 non-null object
host_response_time                4177 non-null object
host_response_rate                4177 non-null object
host_acceptance_rate              3850 non-null object
host_is_superhost                 5820 non-null object
host_thumbnail_url                5820 non-null object
host_picture_url                  5820 non-null object
host_neighbourhood                4977 non-null object
host_listings_count               5820 non-null float64
host_total_listings_count         5820 non-null float64
host_verifications                5835 non-null object
host_has_profile_pic              5820 non-null object
host_identity_verified            5820 non-null object
street                            5835 non-null object
neighbourhood                     4800 non-null object
neighbourhood_cleansed            5835 non-null int64
neighbourhood_group_cleansed      0 non-null float64
city                              5835 non-null object
state                             5835 non-null object
zipcode                           5810 non-null float64
market                            5835 non-null object
smart_location                    5835 non-null object
country_code                      5835 non-null object
country                           5835 non-null object
latitude                          5835 non-null float64
longitude                         5835 non-null float64
is_location_exact                 5835 non-null object
property_type                     5835 non-null object
room_type                         5835 non-null object
accommodates                      5835 non-null int64
bathrooms                         5789 non-null float64
bedrooms                          5829 non-null float64
beds                              5812 non-null float64
bed_type                          5835 non-null object
amenities                         5835 non-null object
square_feet                       302 non-null float64
price                             5835 non-null object
weekly_price                      2227 non-null object
monthly_price                     1717 non-null object
security_deposit                  2770 non-null object
cleaning_fee                      3587 non-null object
guests_included                   5835 non-null int64
extra_people                      5835 non-null object
minimum_nights                    5835 non-null int64
```

```
maximum_nights                      5835 non-null int64
calendar_updated                    5835 non-null object
has_availability                    5835 non-null object
availability_30                     5835 non-null int64
availability_60                     5835 non-null int64
availability_90                     5835 non-null int64
availability_365                    5835 non-null int64
calendar_last_scraped               5835 non-null object
number_of_reviews                   5835 non-null int64
first_review                        3827 non-null object
last_review                         3829 non-null object
review_scores_rating                3789 non-null float64
review_scores_accuracy              3776 non-null float64
review_scores_cleanliness           3778 non-null float64
review_scores_checkin               3778 non-null float64
review_scores_communication         3778 non-null float64
review_scores_location              3779 non-null float64
review_scores_value                 3778 non-null float64
requires_license                    5835 non-null object
license                             1 non-null float64
jurisdiction_names                  0 non-null float64
instant_bookable                    5835 non-null object
cancellation_policy                 5835 non-null object
require_guest_profile_picture       5835 non-null object
require_guest_phone_verification    5835 non-null object
calculated_host_listings_count      5835 non-null int64
reviews_per_month                   3827 non-null float64
dtypes: float64(20), int64(14), object(58)
memory usage: 4.1+ MB
```

It turns out that one record displays a very odd location and, for the sake of the illustration, we will remove it:

```
odd = lst.loc[lst.longitude>-80, ['longitude', 'latitude']]
odd
```

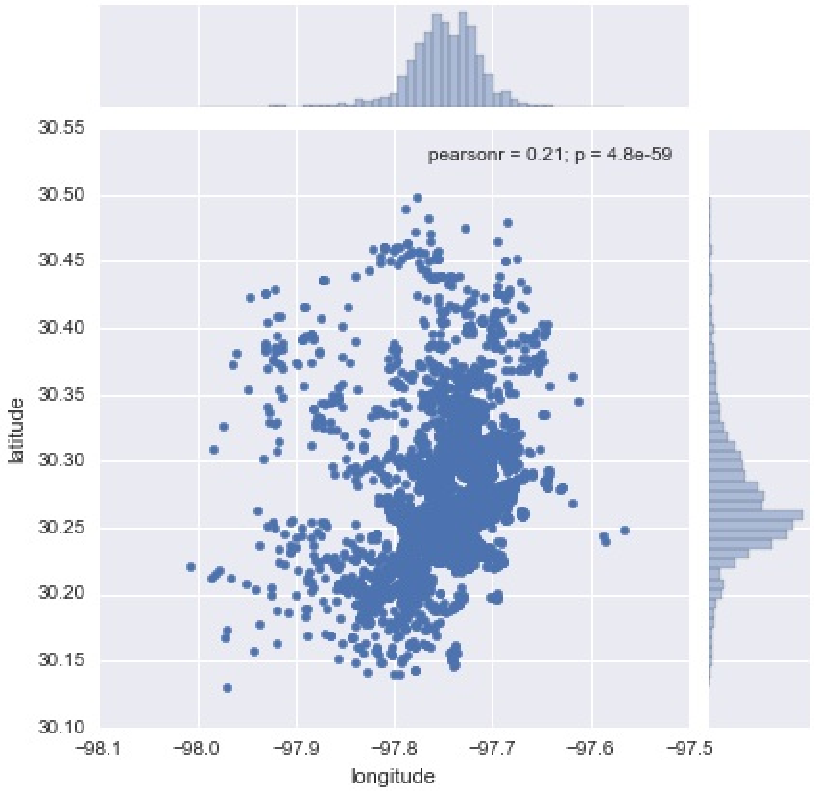
	longitude	latitude
5832	-5.093682	43.214991

```
lst = lst.drop(odd.index)
```

Point Visualization

The most straighforward way to get a first glimpse of the distribution of the data is to plot their latitude and longitude:

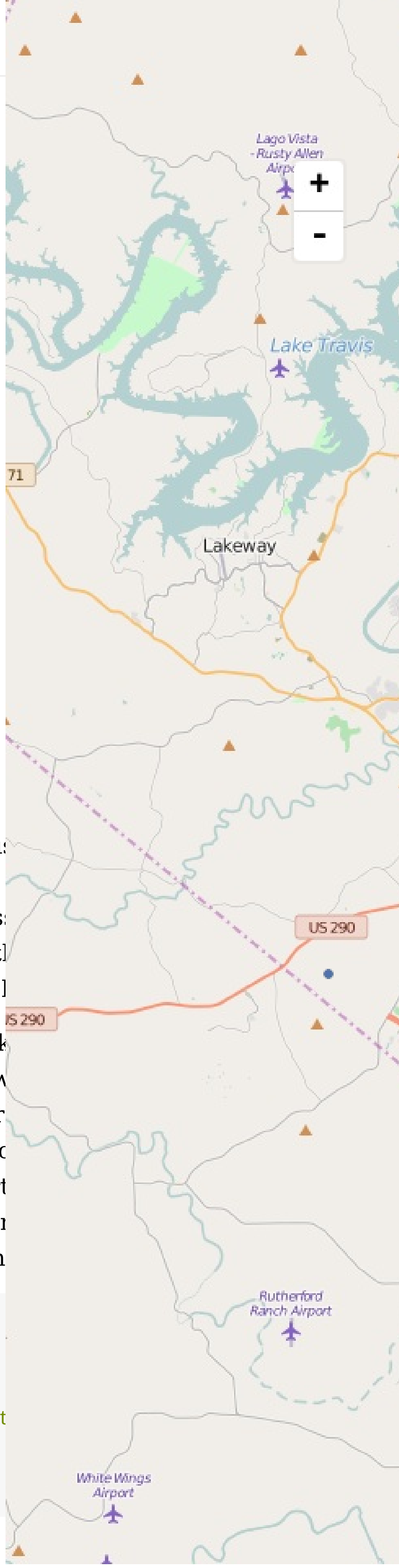
```
sns.jointplot(x="longitude", y="latitude", data=lst);
```

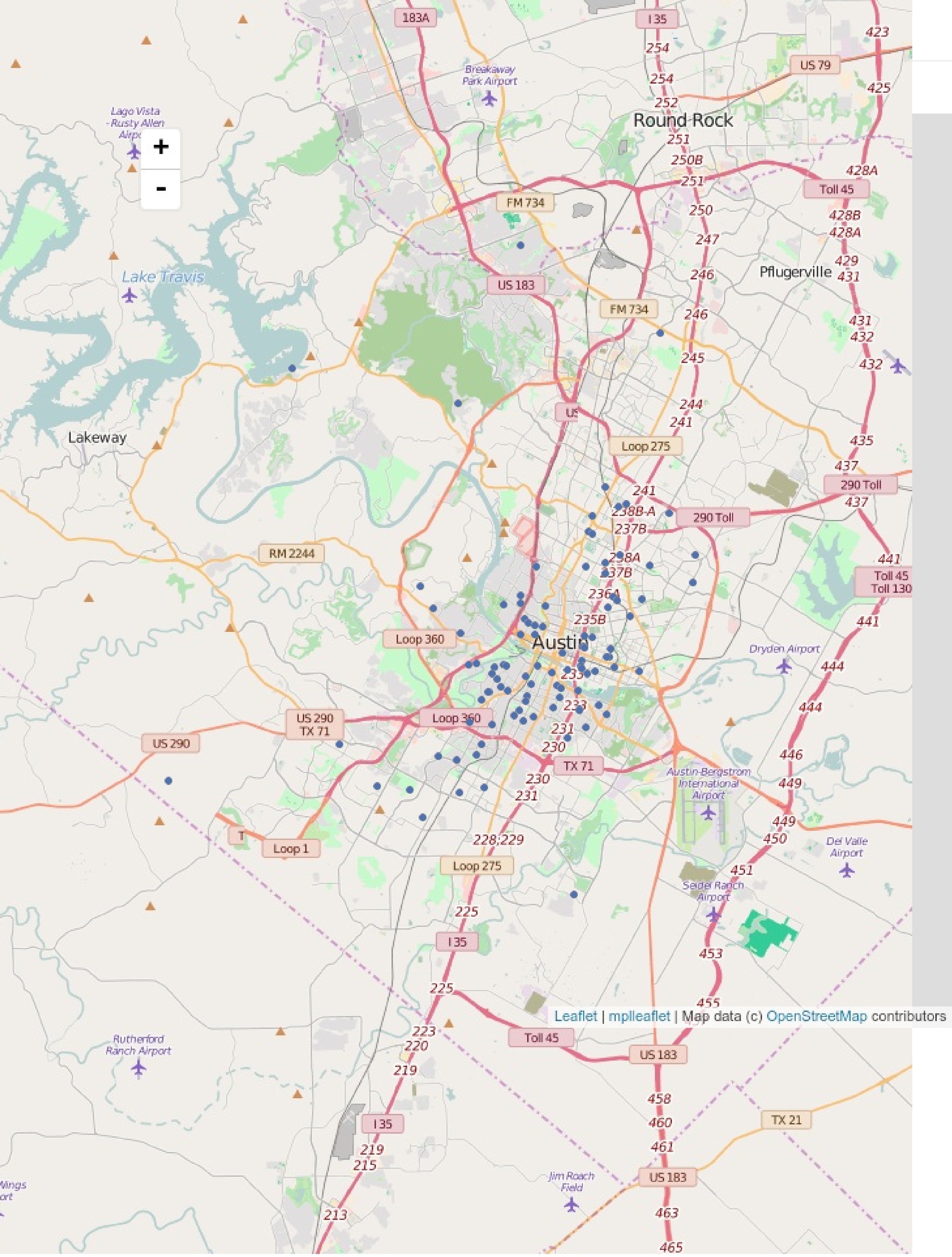


Now this does not neccesarily tell us much about the dataset or the dis
within Austin. There are two main challenges in interpreting the plot:
context, which means the points are not identifiable over space (unless
with lon/lat pairs that they have a clear meaning to you); and two, in th
there are so many points that it is hard to tell any patter other than a l

Let us first focus on the first problem, geographical context. The quick
to provide context to this set of points is to overlay a general map. If w
the map or a set of several data sources that we could aggregate to cr
build it from scratch. But in the XXI Century, the easiest is to overlay o
top of a web map. In this case, we will use Leaflet, and we will convert
`matplotlib` points with `mplleaflet`. The full dataset (+5k observation
for leaflet to plot it directly on screen, so we will obtain a random sam

```
rids = np.arange(lst.shape[0])
np.random.shuffle(rids)
f, ax = plt.subplots(1, figsize=(12, 12))
lst.iloc[rids[:100], :].plot(kind='scatter', x='longitude', y='latit
                             s=30, linewidth=0, ax=ax)
mpll.display(fig=f,)
```




Round Rock
Pflugerville
Austin
Lakeway
Lake Travis
Lago Vista - Rusty Allen Airport
Breakaway Park Airport
Dryden Airport
Austin-Bergstrom International Airport
Del Valle Airport
Seidel Ranch Airport
Rutherford Ranch Airport
Jim Roach Field
183A
I 35
US 79
FM 734
US 183
Toll 45
Loop 275
290 Toll
RM 2244
Loop 360
US 290 TX 71
US 290
TX 71
Loop 1
Toll 45 Toll 130
TX 21
Leaflet | mplleaflet | Map data (c) OpenStreetMap contributors

Spatial Clustering

```
%matplotlib inline

import pandas as pd
```

```
/home/dani/anaconda/envs/pydata/lib/python2.7/site-packages/matplotlib/font_manager.py:
273: UserWarning: Matplotlib is building the font cache using fc-list. This may take a
moment.
  warnings.warn('Matplotlib is building the font cache using fc-list. This may take a m
oment.')
```

```
link = '/home/dani/Desktop/listings.csv'
db = pd.read_csv(link)
db.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5835 entries, 0 to 5834
Data columns (total 92 columns):
id                                    5835 non-null int64
listing_url                           5835 non-null object
scrape_id                             5835 non-null int64
last_scraped                          5835 non-null object
name                                  5835 non-null object
summary                               5373 non-null object
space                                 4475 non-null object
description                           5832 non-null object
experiences_offered                   5835 non-null object
neighborhood_overview                 3572 non-null object
notes                                 2413 non-null object
transit                               3492 non-null object
thumbnail_url                         5542 non-null object
medium_url                            5542 non-null object
picture_url                           5835 non-null object
xl_picture_url                        5542 non-null object
host_id                               5835 non-null int64
host_url                              5835 non-null object
host_name                             5820 non-null object
host_since                            5820 non-null object
host_location                         5810 non-null object
host_about                            3975 non-null object
host_response_time                    4177 non-null object
host_response_rate                    4177 non-null object
host_acceptance_rate                  3850 non-null object
host_is_superhost                     5820 non-null object
host_thumbnail_url                    5820 non-null object
host_picture_url                      5820 non-null object
```

```
host_neighbourhood                    4977 non-null object
host_listings_count                   5820 non-null float64
host_total_listings_count             5820 non-null float64
host_verifications                    5835 non-null object
host_has_profile_pic                  5820 non-null object
host_identity_verified                5820 non-null object
street                                5835 non-null object
neighbourhood                         4800 non-null object
neighbourhood_cleansed                5835 non-null int64
neighbourhood_group_cleansed          0 non-null float64
city                                  5835 non-null object
state                                 5835 non-null object
zipcode                               5810 non-null float64
market                                5835 non-null object
smart_location                        5835 non-null object
country_code                          5835 non-null object
country                               5835 non-null object
latitude                              5835 non-null float64
longitude                             5835 non-null float64
is_location_exact                     5835 non-null object
property_type                         5835 non-null object
room_type                             5835 non-null object
accommodates                          5835 non-null int64
bathrooms                             5789 non-null float64
bedrooms                              5829 non-null float64
beds                                  5812 non-null float64
bed_type                              5835 non-null object
amenities                             5835 non-null object
square_feet                           302 non-null float64
price                                 5835 non-null object
weekly_price                          2227 non-null object
monthly_price                         1717 non-null object
security_deposit                      2770 non-null object
cleaning_fee                          3587 non-null object
guests_included                       5835 non-null int64
extra_people                          5835 non-null object
minimum_nights                        5835 non-null int64
maximum_nights                        5835 non-null int64
calendar_updated                      5835 non-null object
has_availability                      5835 non-null object
availability_30                       5835 non-null int64
availability_60                       5835 non-null int64
availability_90                       5835 non-null int64
availability_365                      5835 non-null int64
calendar_last_scraped                 5835 non-null object
number_of_reviews                     5835 non-null int64
first_review                          3827 non-null object
last_review                           3829 non-null object
review_scores_rating                  3789 non-null float64
review_scores_accuracy                3776 non-null float64
review_scores_cleanliness             3778 non-null float64
review_scores_checkin                 3778 non-null float64
review_scores_communication           3778 non-null float64
```

```
review_scores_location              3779 non-null float64
review_scores_value                 3778 non-null float64
requires_license                    5835 non-null object
license                             1 non-null float64
jurisdiction_names                  0 non-null float64
instant_bookable                    5835 non-null object
cancellation_policy                 5835 non-null object
require_guest_profile_picture       5835 non-null object
require_guest_phone_verification    5835 non-null object
calculated_host_listings_count      5835 non-null int64
reviews_per_month                   3827 non-null float64
dtypes: float64(20), int64(14), object(58)
memory usage: 4.1+ MB
```

Spatial Regression