Tutorial: Stochastic Geometry / Spatial Point Processes

Note

This tutorial aims to simulate different spatial point processes.

1 Poisson processes

This process simulates a conditional set of $point_{nb}$ points in a spatial domain D defined by the values $x_{min}, x_{max}, y_{min}, y_{max}$. In order to simulate a non conditional Poisson point process of intensity λ within a domain S, it is necessary to generate a random number of points according to a Poisson law with the parameter λS : $point_{nb} = poisson(\lambda S)$ (i.e. the number of points is a random variable following a Poisson Law).

The coordinates of each point follow a uniform distribution.

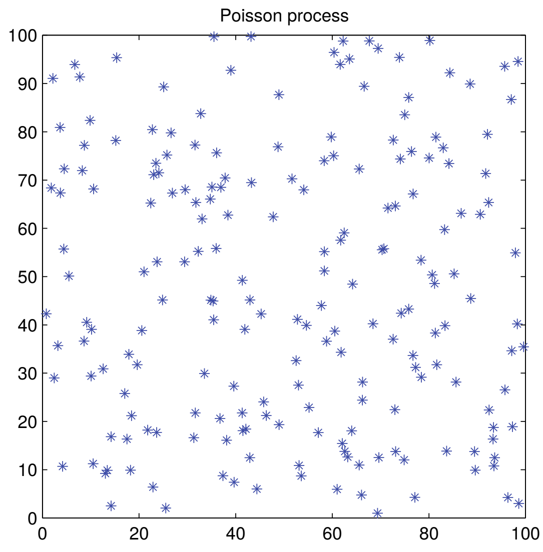

Figure 1: Conditional Poisson point process, with 200 points.

?

1. Simulate a conditional Poisson point process on a spatial domain D with a fixed number of points (see Fig. 1).
2. Simulate several distributions.

The function poissrnd can be used to generate and random variable following a Poisson law. Use rand for generating uniform distribution random variables.

Import scipy.stats to use the function poisson and np.random for more classical stochastic functions.

2 Neyman-Scott processes

This process simulates aggregated sets of points within a spatial domain D defined by the values $x_{min}, x_{max}, y_{min}, y_{max}$. For each aggregate ($n_{par}$), we first generate the random position of the 'parent' point. Then, the 'children' points are simulated in a neighborhood (within a square box of size r^2_{child}) of the 'parent' point. The number of points for each aggregate is either fixed or randomized according to a Poisson law of parameter n_{child} (see Fig. 2).

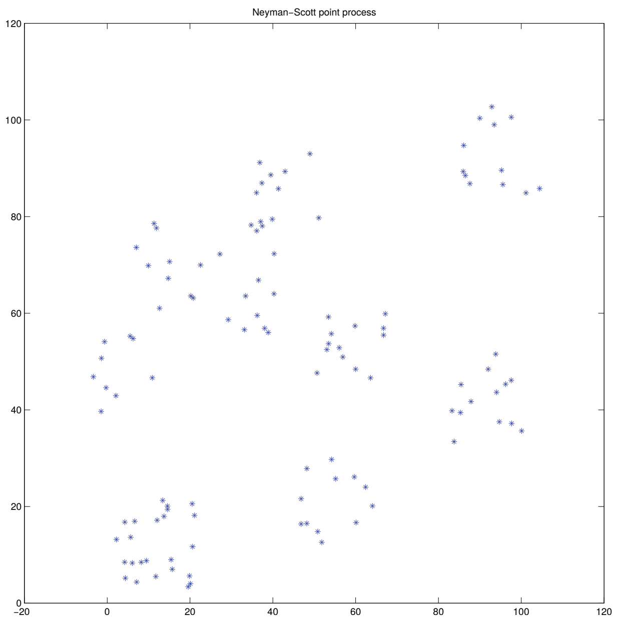

Figure 2: Neyman-Scott point process with $h_{child} = 10$ and $n_{par} = 10$

?

1. Simulate a process of 3 aggregates with 10 points.
2. Simulate a process of 10 aggregates with 5 points.

3 Gibbs processes

The idea of Gibbs processes is to spatially distribute the points according to some laws of interactions (attraction or repulsion) within a variable range.

Such a process can be defined by a cost function $f(d)$ that represents the cost associated to the presence of 2 separated points by a distance d (see Fig. 3). For a fixed value r, if $f(d)$ is negative, there is a high probability to find 2 points at a distance d (attraction). Conversely, if $f(d)$ is positive, there is a weak probability to find 2 points at a distance d (repulsion).

- Code such a function, with prototype `function e=f(d)` or `def energy(d):`, where

$$f(d) = \begin{cases} 50 & \text{if } 0 < d <= 15 \\ -10 & \text{if } 15 < d <= 30 \\ 0 & otherwise \end{cases}$$

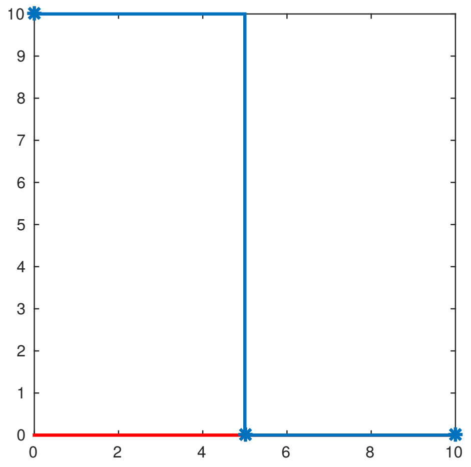

Figure 3: Energy function f.

The generation process reorganizes an initial Poisson point process within the spatial domain according to a specific cost piecewise constant function. The reorganization consists in a loop process of nb_{iter} iterations. For each iteration, we calculate the total energy:

$$e = \sum_{(i,j),i \neq j} f(dist(x_i, x_j)) \tag{1}$$

The objective is to minimize this energy. For doing that, for each iteration we replace a point by 4 (for example) random points and we calculate the energy for each configuration. The initial point is either preserved or replaced (by one of the four points) according to the minimal energy.

?

1. Code a function with the following prototype in MATLAB® (the energyFunction is previously noticed f). It computes the energy between all the points present in $[x, y]$ and point $[x_k, y_k]$. The energyFunction converts a distance to an 'energy', reflecting attractivity or repulsivity.

```matlab
function e = energy(energyFunction, x, y, xk, yk)
```

```python
def energy(P, eFunction=exampleEnergyFunction):
```

2. Simulate a regular point process by choosing a specific energy function.
3. Change the cost function and simulate a few aggregated point processes (see Fig. 4).

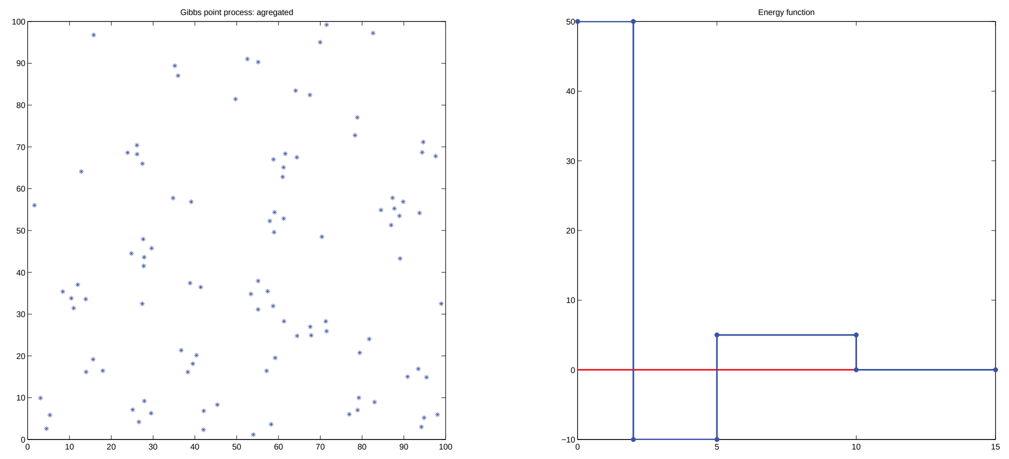

Figure 4: Gibbs agregated point process and its energy function.

4 Ripley function

The Ripley function $K(r)$ characterizes the spatial distribution of the points. For a Poisson process of density λ, $\lambda K(r)$ is equal to the mean value of the number of neighbors at a distance lower than r to any point. In the case where the process is not known (λ?), the Ripley function has to be estimated (approximated) with the unique known realization. It is the first estimator of $K(r)$:

$$K(r) = \frac{1}{\lambda}\frac{1}{N}\sum_{i=1}^{N}\sum_{i \neq j} k_{ij} \tag{2}$$

where N is the number of points in the studied domain D, $\lambda = N/D$ is the estimator of the density of the process and k_{ij} takes the value 1 if the distance between the point i and the point j is lower than r, and 0 in the other case.

We denote:

$$L(r) = \sqrt{K(r)/\pi} \tag{3}$$

?

1. Code a function to compute K (`function K=ripley(points, box, r)`), with `points` being the considered points, `box` the domain, and r the distance.
2. Calculate the Ripley function for an aggregated point process, a Poisson point process and a regular point process.
3. Display the value $L(r) - r$ as in Fig 5.

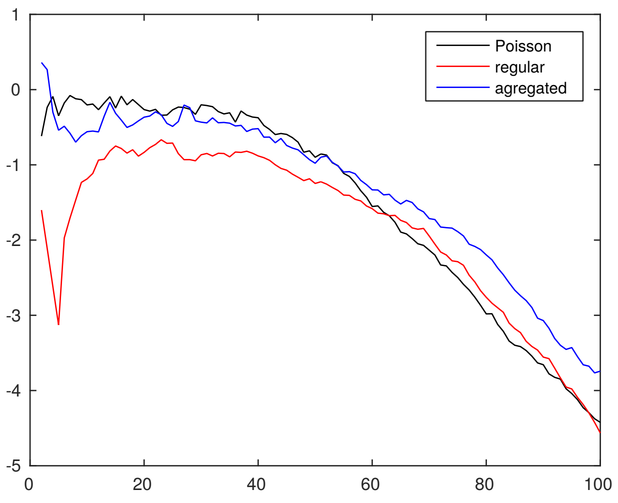

Figure 5: Ripley's function $L(r) - r$.

5 Marked point processes

To simulate a complex point process, it can be useful to associate several random variables (marks) for each point.

1. Simulate a disk process, where the points (disk centers) are defined according to a Poisson process and the radii are selected with a Gaussian law.
2. Add a second mark for allocating a color to each disk (uniform law).

An example result is shown in Fig. 6

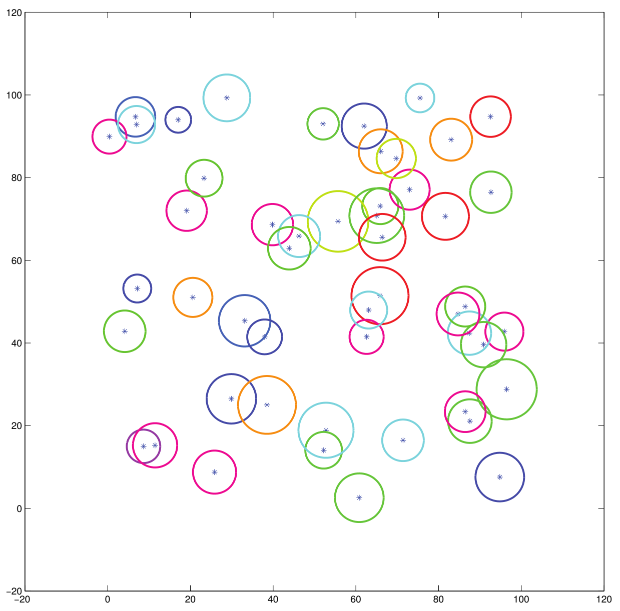

Figure 6: A Poisson point process is used to generates the center of the circles. The radii are chosen according a Gaussian law, and the color according a uniform law.