1 Question 1 (10 points)

1. Code a function `Count_Pairs(X,R)` that counts the number of pairs of points in a set X located at a distance less than R from each other.

We want to implement a function that returns a Strauss-like point process. A Strauss process is based on iterations upon a constrained[1] Poisson process in a window of size $[0;1] \times [0;1]$, where X_0 will denote the initial set of points. At each iteration, one of the points is slightly moved (e.g. according to a normal distribution of low variance). We then consider the quantity α defined as follows.

- If the point leaves the observation window, it returns to its initial position, i.e. $\alpha = 0$.
- If `Count_Pairs(`X_{n+1}`,R)` is less than `Count_Pairs(`X_n`,R)`, with X_n the set of points at the iteration n and R a parameter of the process, then $\alpha = 1$.
- Else $\alpha = \gamma^{(\texttt{Count\_Pairs}(X_{n+1},R) - \texttt{Count\_Pairs}(X_n,R))}$, with γ a parameter of the process.
- Finally, a number β is drawn randomly between 0 and 1. If $\beta < \alpha$, then the point set is updated with the shifted element. Otherwise, the original set is retained.

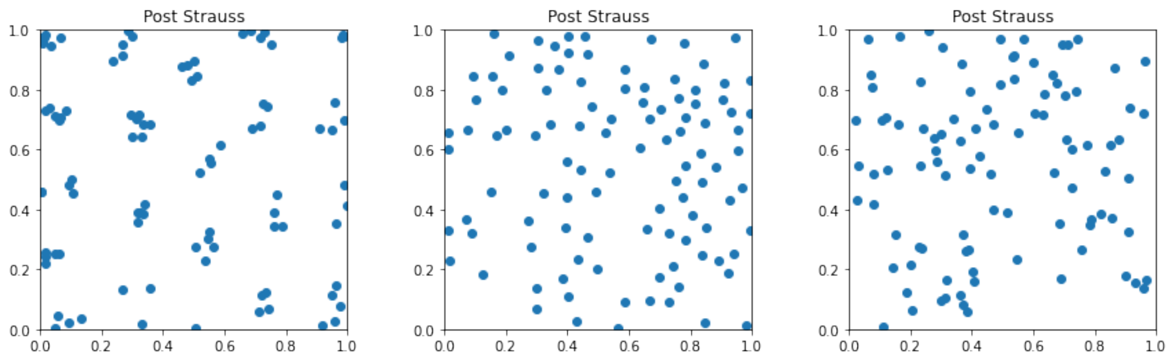
2. Code a function `Strauss(n,gamma,R,N)` that generates a Strauss point process of n points and with parameters $\gamma \in [0;1]$, $R > 0$ and N the number of iterations.

a. Test your `Strauss(n,gamma,R,N)` function with $n = 100$, $\gamma = 0.1$, $R = 0.2$ and $N \geq 1000$.

b. Test your `Strauss(n,gamma,R,N)` function with $n = 100$, $\gamma = 0.1$, $R = 0.05$ and $N \geq 1000$.

c. What happens if $\gamma = 1$?

The possible outcomes of questions **a**, **b** and **c** are illustrated below.



3. How could you both quantitatively and qualitatively characterise the different point process you can generate with this function? (Bonus: do it.)

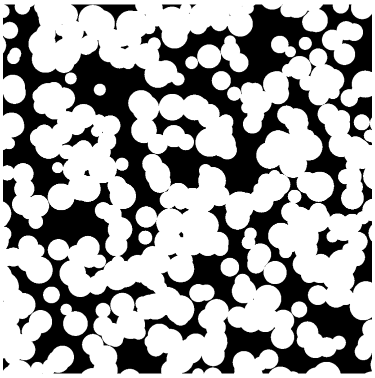
Useful Python function: `scipy.spatial.distance.pdist`.

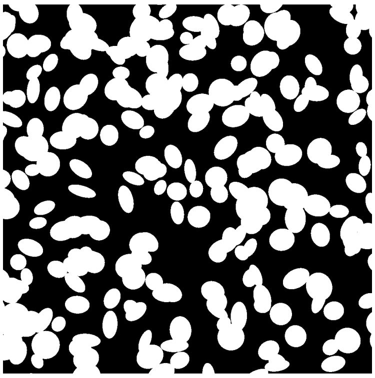
Useful MATLAB function : `pdist`.

[1] A constrained Poisson process has a fixed number of points which doesn't follow on a Poisson law anymore.

2 Question 2 (10 points)

Let us consider a set of binary images of size 1024×1024 made of white particles on a black background. The particles are disks or ellipses of random radius.





Propose and implement a method to retrieve the characteristics of the mean particle for each set of images at your disposal: area, perimeter and mean radius for disks; mean equivalent radius for ellipses.

Explain your methodology and any assumptions you may have to make.

Miles formulas in 2-D:

$$\frac{A}{W_{size}} = 1 - e^{-\lambda a} \tag{1}$$

$$\frac{P}{W_{size}} = e^{-\lambda a} \times \lambda p \tag{2}$$

$$\frac{\pi\chi}{W_{size}} = e^{-\lambda a}\left(\pi\lambda x - \frac{1}{4}(\lambda p)^2\right) \tag{3}$$

where A, P and χ are the expected value of area, perimeter and euler's number measured on the images, and a, p and x their counterpart for the mean particle. The parameter λ would be the density of particles and W_{size} the area of the observation window.

Useful Python functions: `skimage.measure.area`, `skimage.measure.perimeter`, `skimage.measure.euler_number`.
Optional, yet useful, Python function: `scipy.optimize.fmin`.

Useful MATLAB functions: `bwarea`, `bwperim`, `bweuler`.
Optional, yet useful, MATLAB function: `fminsearch`.