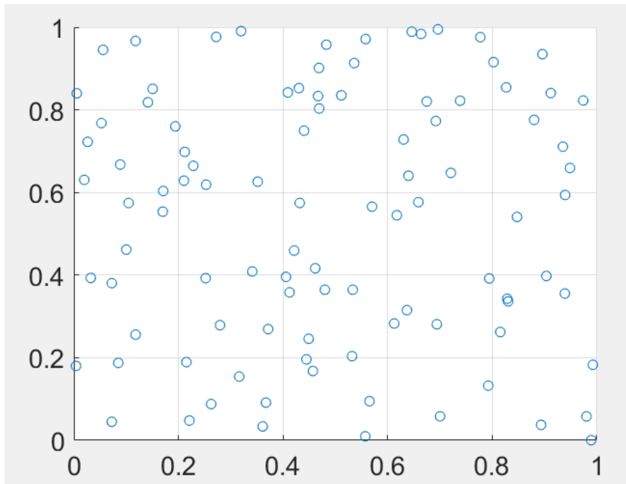
Nom : Wilmar QUIROGA

RIO208 - TP TOPOLOGIE ALGÉBRIQUE

1) Simuler un processus de Poisson d'intensité $\lambda = 100$ sur un carré de côté a = 1.



2) Construire le 2-squelette du complexe simplicial de Rips-Vietoris de paramètre $\epsilon = 0.1$ sur le processus de Poisson précédemment simulé, i.e. calculer et stocker les listes des 0-simplexes, des 1-simplexes, et des 2-simplexes. Afficher les dans une figure.

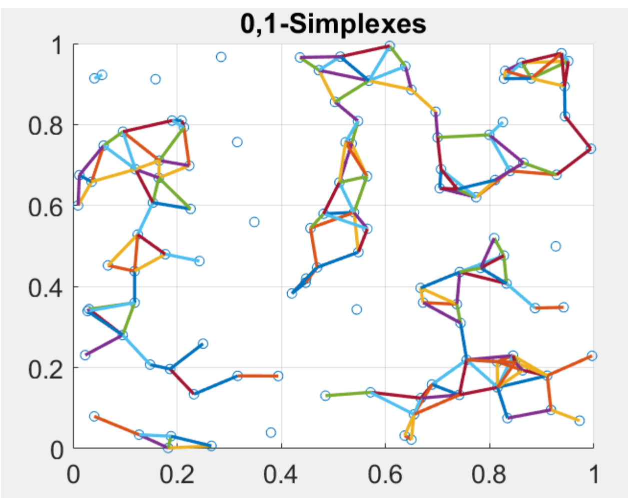
Pour le calcul des 1-simplexes on crée deux matrices pour les coordonnées X et Y de dimension Nx2 dans lesquelles on stocke la position des points initiaux et finaux qui composent chaque 1-simplexe, puis pour la construction on vérifie que la distance entre les points est inférieure à 0.1.

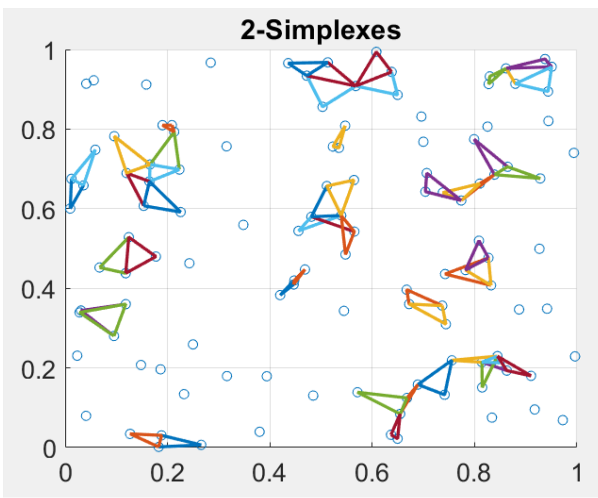
```
areaTotal=1*1;
lambda=100; %intensity km^-1
numbPoints=poissrnd(areaTotal*lambda);%Nombre des Points Poisson.
xx=xCarre*(rand(numbPoints,1));
yy=yCarre*(rand(numbPoints,1));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% 1-Simplex
Simplex1x=zeros(1,2);
Simplex1y=zeros(1,2);
count=1;
for i=1:1:numbPoints
   X=[xx(i) yy(i)];
   for j=i+1:1:numbPoints
       if (j>i)
           Y=[xx(j) yy(j)];
           distance=pdist2(X,Y);
           if(distance<0.1)
               Simplex1x(count,1)=xx(i);
               Simplex1y(count,1)=yy(i);
               Simplex1x(count,2)=xx(j);
               Simplex1y(count,2)=yy(j);
               count=count+1;
           end
       end
   end
end
```

Pour le calcul des 2-simplexes on crée deux matrices pour les coordonnées X et Y de dimension Nx3 dans lesquelles on stocke la position des trois points qui forment un triangle, pour la construction des 2-simplexes, on crée une boucle qui passe par les 1-simplexes et ensuite, une autre qui passe par les 0-simplexes, et on vérifie que la distance des deux points du 1-simplexe et d'un point du 0-simplexe est inférieure à 0.1 en même temps.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% 2-Simplex
Simplex2x=zeros(1,3);
Simplex2y=zeros(1,3);
count2=1;
Simplex1xLen=count-1;
for i=1:1:Simplex1xLen
  X=[Simplex1x(i,1) Simplex1y(i,1)];
  Y=[Simplex1x(i,2) Simplex1y(i,2)];
  for j=1:1:numbPoints
         Z=[xx(j) yy(j)];
         distance2=pdist2(X,Z);
         distance3=pdist2(Y,Z);

         if (X(1)~=Z(1)&&X(2)~=Z(2))&&(Y(1)~=Z(1)&&Y(2)~=Z(2))&&(isempty(find(Simplex2x==Z(1))))
           if distance2<0.1 && distance3<0.1
             Simplex2x(count2,1)=Simplex1x(i,1);
             Simplex2y(count2,1)=Simplex1y(i,1);
             Simplex2x(count2,2)=Simplex1x(i,2);
             Simplex2y(count2,2)=Simplex1y(i,2);
             Simplex2x(count2,3)=xx(j);
             Simplex2y(count2,3)=yy(j);
             Simplex2x(count2,4)=Simplex1x(i,1);
             Simplex2y(count2,4)=Simplex1y(i,1);
             count2=count2+1;
           end
         end
  end
end
```





3) Calculer les matrices ∂_1 et ∂_2

Pour le calcul du matrix ∂_1 de taille #0-simplexes x #1-Simplexes, On place à chaque position j,i de la matrice ∂_1 , un -1 ou un 1, si le 0-simplex fait partie du 1-simplex, et 0 si le point du 0-

simplex n'est pas contenu dans le 1-simplex, on place un 1 ou un -1 selon l'orientation de construction du 1-simplex.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Calcul Matrux d1
Simplex2xLen=count2-1;
d1=zeros(numbPoints,Simplex1xLen);
for i=1:1:Simplex1xLen
   Px1=[Simplex1x(i,1) Simplex1y(i,1)];
   Px2=[Simplex1x(i,2) Simplex1y(i,2)];
   for j=1:1:numbPoints
       X=[xx(j,1) yy(j,1)];
       if Px1(1)==X(1) &&  Px1(2)==X(2)
          d1(j,i)=-1;
       end
       if Px2(1)==X(1) &&  Px2(2)==X(2)
          d1(j,i)=1;
       end
   end
end
```

Pour le calcul du matrix ∂_2 de taille #1-simplexes x #2-Simplexes, On place à chaque position j,i de la matrice ∂_2 , un -1 ou un 1, si le 1-simplex fait partie du 2-simplex. Étant donné que chaque 2-simplexe est construit à partir de 3 1-simplexes, il y aura trois valeurs différentes de 0 dans chaque colonne de la matrice ∂_2 . D'autre part, on met un 1 ou -1 dans la position j,i, selon le sens de construction du 2-simple.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Calcul Matrux d2
d2=zeros(Simplex1xLen,Simplex2xLen);
for i=1:1:Simplex2xLen
   Px1=[Simplex2x(i,1) Simplex2y(i,1)];
   Px2=[Simplex2x(i,2) Simplex2y(i,2)];
   Px3=[Simplex2x(i,3) Simplex2y(i,3)];

   for j=1:1:Simplex1xLen
       X1=[Simplex1x(j,1) Simplex1y(j,1)];
       X2=[Simplex1x(j,2) Simplex1y(j,2)];
       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Px1->Px2
       if Px1(1)==X1(1) && Px1(2)==X1(2) && Px2(1)==X2(1) && Px2(2)==X2(2)
          d2(j,i)=1;
       end
       if Px1(1)==X2(1) && Px1(2)==X2(2) && Px2(1)==X1(1) && Px2(2)==X1(2)
          d2(j,i)=-1;
       end
       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Px2->Px3
       if Px2(1)==X1(1) &&  Px2(2)==X1(2) && Px3(1)==X2(1) &&  Px3(2)==X2(2)
          d2(j,i)=1;
       end
       if Px2(1)==X2(1) &&  Px2(2)==X2(2) && Px3(1)==X1(1) &&  Px3(2)==X1(2)
          d2(j,i)=-1;
       end
       %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Px3->Px1
       if Px3(1)==X1(1) &&  Px3(2)==X1(2) && Px1(1)==X2(1) &&  Px1(2)==X2(2)
          d2(j,i)=1;
       end
       if Px3(1)==X2(1) &&  Px3(2)==X2(2) && Px1(1)==X1(1) &&  Px1(2)==X1(2)
          d2(j,i)=-1;
       end
   end
end
```

4) Calculer les nombres de Betti β_0 et β_1

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% B0,B1,Xc
B0=numbPoints-rank(d1);
[f, g]=size(null(d1));
B1=g-rank(d2);
Xc=numbPoints-Simplex1xLen+Simplex2xLen;
```

$$\beta_0 \quad = \quad \dim\ker\partial_0 - \dim\operatorname{im}\partial_1$$

$$\beta_0 = 104 - 90 = 14$$

$$\beta_1 \quad = \quad \dim\ker\partial_1 - \dim\operatorname{im}\partial_2$$

$$\beta_1 = 85 - 49 = 36$$

5) Calculer la caractéristique d'Euler x

$$\chi(C) = \sum_{i=0}^{\infty}(-1)^i s_i$$

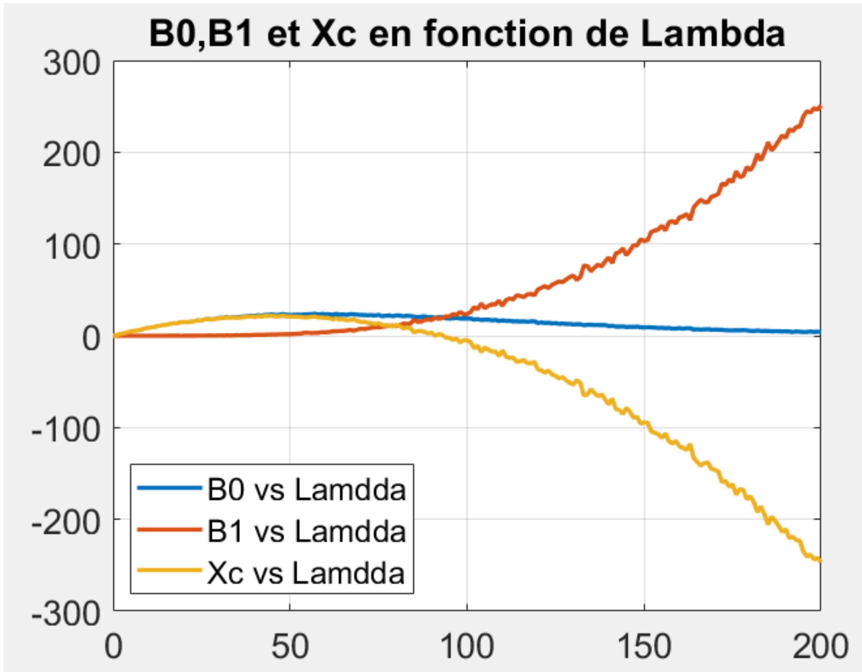
$$x = s_0 - s_1 + s_2 = 104 - 175 + 49 = -22 = \beta_0 - \beta_1$$

6) Tracer, β_0 , β_1 et X en fonction de λ. Faire varier λ de 0 à 200 en faisant la moyenne sur au moins 100 simulations par valeur de λ.

On crée une fonction ***SimplexCalcul(****Lambda****)*** qui reçoit comme paramètres Lambda et calcule β_0 , β_1 et X pour chaque itération de cette façon, on peut faire le graphique de β_0 , β_1 et X et pour toutes les valeurs de Lambda, et pour chaque valeur de Lambda faire la moyenne de la valeur de β_0 , β_1 et X sur 100 itérations.

```
%%%% Vecteurs Lambda,B0,B1,Xc pour Tracer B0,B1 et Xc en fonction de Lambda
VecLambda=linspace(0,200,201);
VecB0=zeros(1,length(VecLambda));
VecB1=zeros(1,length(VecLambda));
VecXc=zeros(1,length(VecLambda));
%%%%%%%%%%%%%%%%%%%%%Moyenne
N=100; %Nombre Points pour faire la moyenne
VecB0Moy=zeros(1,N);
VecB1Moy=zeros(1,N);
VecXcMoy=zeros(1,N);
%%%%%%%%%%%%%%% moyenne sur 100 simulations par valeur de Lambda
for i=1:1:length(VecLambda)
    for j=1:1:length(VecB0Moy)
        [VecB0Moy(j),VecB1Moy(j),VecXcMoy(j)]=SimplexCalcul(VecLambda(i));
    end
    VecB0(i)=mean(VecB0Moy);
    VecB1(i)=mean(VecB1Moy);
    VecXc(i)=mean(VecXcMoy);
end

figure(2)
plot(VecLambda,VecB0);
hold on
plot(VecLambda,VecB1);
plot(VecLambda,VecXc);
hold off
title('B0,B1 et Xc en fonction de Lambda')
legend({'B0 vs Lamdda','B1 vs Lamdda','Xc vs Lamdda'},'Location','southwest')
```



7) Commenter les courbes.

Pour la courbe de $\boldsymbol{\beta_0}$: On peut voir β_0 comme le nombre de 0-simplexes qui ne limitent pas les 1-simplexes, on voit que lorsque Lambda croît, tous les 0-simplexes limitent les 1-simplexes, ce qui fait que B0 tend vers zéro pour de très grandes valeurs de lambda.

Pour la courbe de $\boldsymbol{\beta_1}$: On peut voir β_1 comme le nombre de trous de couverture, on voit que plus Lambda croît, plus β_1 croît, c'est-à-dire le nombre de cycles de 1-simplexes qui ne sont pas liés. 2-simplexes augmente, donc le nombre de trous de couverture.

Pour la courbe de $\boldsymbol{X_c}$: C'est la différence $\beta_0 - \beta_1$, Comme β_0 tend vers 0 et que β_1 croît, nous pouvons voir que la courbe de Xc décroît presque au même rythme que la courbe de β_1 croît.