Exercises for Spatial Point Processes

Spatial Epidemiologi Autumn 2021

Alexander, Victor, Mikel

11 januar, 2022

Contents

Task 1

Data import and cleaning

```
nucli.23=read.table(file="nucli23.txt",header=T,sep="\t")
head(nucli.23)
```

```
##     NUCLI niu data_pos        X       Y
## 111    23 276       10 300855.2 4494651
## 112    23 294       12 300896.9 4494582
## 113    23 295       12 300871.4 4494621
## 114    23 296       12 300868.3 4494634
## 115    23 623       16 300904.1 4494558
## 116    23 624       16 300902.5 4494569
```

```
summary(nucli.23)
```

```
##      NUCLI         niu           data_pos           X
```

```
##  Min.   :23   Min.   :276.0   Min.   :10.00   Min.   :300845
##  1st Qu.:23   1st Qu.:627.8   1st Qu.:16.00   1st Qu.:300870
##  Median :23   Median :681.5   Median :18.00   Median :300884
##  Mean   :23   Mean   :702.8   Mean   :18.17   Mean   :300881
##  3rd Qu.:23   3rd Qu.:796.2   3rd Qu.:20.00   3rd Qu.:300898
##  Max.   :23   Max.   :965.0   Max.   :22.00   Max.   :300905
##        Y
##  Min.   :4494557
##  1st Qu.:4494580
##  Median :4494599
##  Mean   :4494603
##  3rd Qu.:4494627
##  Max.   :4494651
```

```
poly.23=read.table(file=paste("poly23.txt"),header=T,sep="\t")
summary(poly.23)
```

```
##        X               Y               NUCLI
##  Min.   :300839   Min.   :4494549   Min.   :23
##  1st Qu.:300860   1st Qu.:4494573   1st Qu.:23
##  Median :300878   Median :4494601   Median :23
##  Mean   :300879   Mean   :4494603   Mean   :23
##  3rd Qu.:300900   3rd Qu.:4494632   3rd Qu.:23
##  Max.   :300915   Max.   :4494657   Max.   :23
```

```
# Change reference of X and Y (to 0).
min.X=min(nucli.23$X)
min.Y=min(nucli.23$Y)
nucli.23$X=nucli.23$X-min.X
nucli.23$Y=nucli.23$Y-min.Y
summary(nucli.23)
```

```
##      NUCLI         niu           data_pos          X               Y
##  Min.   :23   Min.   :276.0   Min.   :10.00   Min.   : 0.00   Min.   : 0.00
##  1st Qu.:23   1st Qu.:627.8   1st Qu.:16.00   1st Qu.:24.20   1st Qu.:23.77
##  Median :23   Median :681.5   Median :18.00   Median :39.01   Median :42.12
##  Mean   :23   Mean   :702.8   Mean   :18.17   Mean   :35.99   Mean   :46.13
##  3rd Qu.:23   3rd Qu.:796.2   3rd Qu.:20.00   3rd Qu.:52.33   3rd Qu.:70.55
##  Max.   :23   Max.   :965.0   Max.   :22.00   Max.   :59.82   Max.   :94.80
```

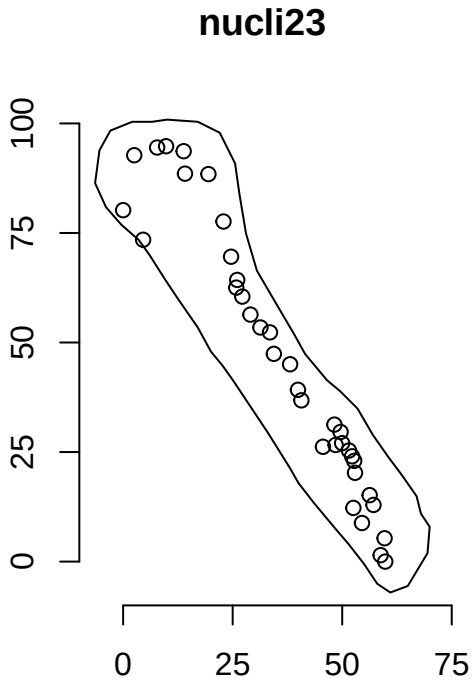
```
poly.23$X=poly.23$X-min.X
poly.23$Y=poly.23$Y-min.Y
summary(poly.23)
```

```
##        X               Y              NUCLI
##  Min.   :-6.403   Min.   : -7.06   Min.   :23
##  1st Qu.:14.813   1st Qu.: 16.42   1st Qu.:23
##  Median :33.033   Median : 44.91   Median :23
##  Mean   :33.511   Mean   : 46.86   Mean   :23
##  3rd Qu.:54.248   3rd Qu.: 75.89   3rd Qu.:23
##  Max.   :69.972   Max.   :100.87   Max.   :23
```

1.1) Build a ppp object using the "poly23" data as a window.

```
pol.illa<-list(x=poly.23$X,y=poly.23$Y)
n23=ppp(nucli.23$X,nucli.23$Y,poly=pol.illa)
```

```
plot(n23,main="nucli23")
axis(1,at=c(seq(0,75,by=25)),pos=c(-10,0))
axis(2,at=c(seq(0,100,by=25)),pos=c(-10,-0))
```



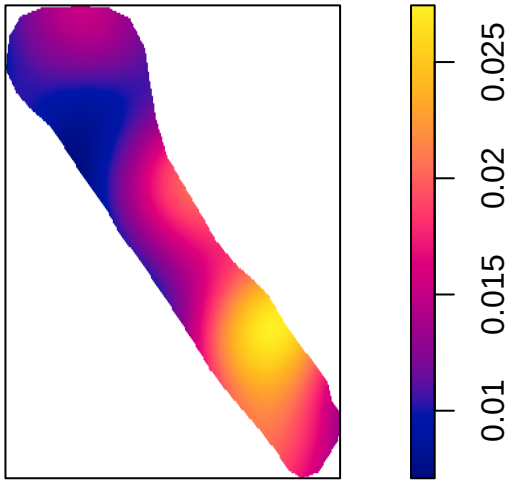
1.2) Briefly describe the point pattern process.

```
summary(n23)
```

```
## Planar point pattern:  36 points
## Average intensity 0.01451131 points per square unit
##
## Coordinates are given to 6 decimal places
##
## Window: polygonal boundary
## single connected closed polygon with 47 vertices
## enclosing rectangle: [-6.40267, 69.97237] x [-7.06032, 100.8694] units
##                      (76.38 x 107.9 units)
## Window area = 2480.82 square units
## Fraction of frame area: 0.301
```

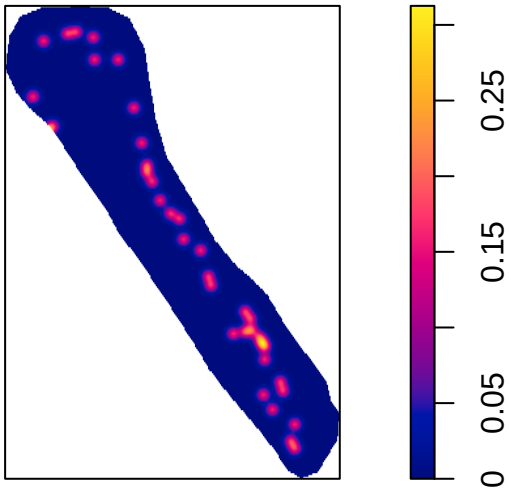
```
n23.den<-density(n23,dimyx=c(256,256))
plot(n23.den)
```

n23.den



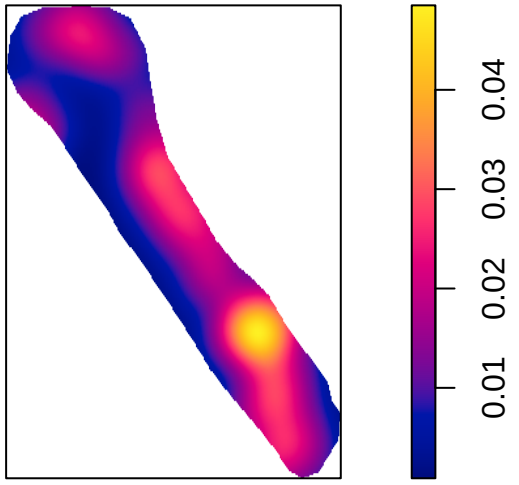

```
plot(density(n23, dimyx=c(256,256), sigma=1))
```

density(n23, dimyx = c(256, 256), sigma = 1)



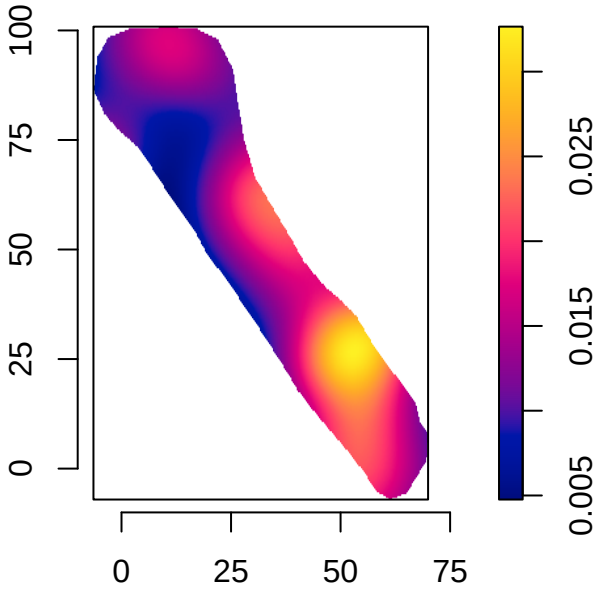

```
plot(density(n23, dimyx=c(256,256), sigma=5))
```

density(n23, dimyx = c(256, 256), sigma = 5)



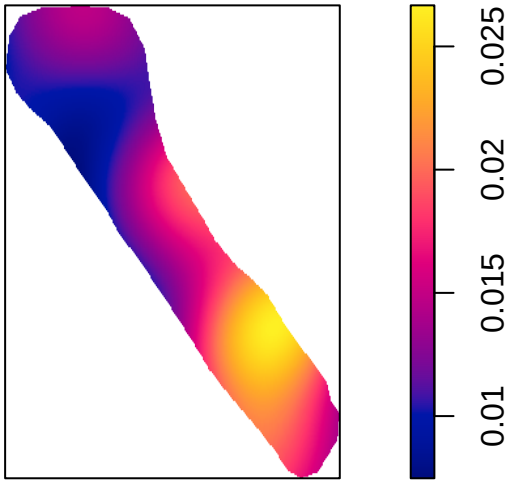

```
plot(density(n23, dimyx=c(256,256), sigma=7.5))
axis(1,at=c(seq(0,75,by=25)),pos=c(-10,0))
axis(2,at=c(seq(0,100,by=25)),pos=c(-10,-0))
```

density(n23, dimyx = c(256, 256), sigma = 7.5)



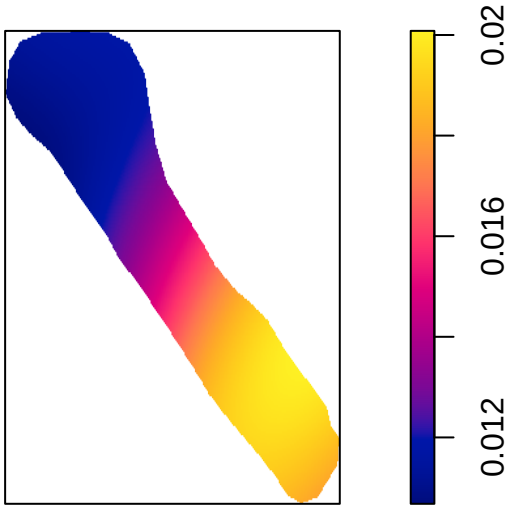

```
plot(density(n23, dimyx=c(256,256), sigma=10))
```

density(n23, dimyx = c(256, 256), sigma = 10)



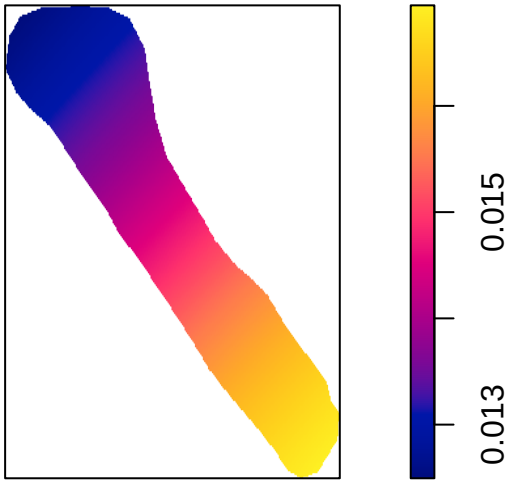

```
plot(density(n23, dimyx=c(256,256), sigma=20))
```

density(n23, dimyx = c(256, 256), sigma = 20)




```
plot(density(n23, dimyx=c(256,256), sigma=50))
```

density(n23, dimyx = c(256, 256), sigma = 50)



We can see that sigma = 20 and sigma = 50 gives too much smoothing. Similarly, we can see that sigma = 1 gives to little smoothing. The density with sigma between 5 and 10 looks more informative.

Additionally, we can see that the process is not homogeneous. There are some regions that clearly have a larger intensity of points. The regions around $(x, y) = (60, 25)$ and $(x, y) = (30, 60)$ are two examples, where the first one has the largest intensity.

1.3) Rebuild the ppp object using the "time to nest" (data_pos variable) as marks.

```
(n23T=ppp(nucli.23$X,nucli.23$Y,poly=pol.illa,marks=nucli.23$data_pos))
```

```
## Marked planar point pattern: 36 points
## marks are numeric, of storage type  'integer'
## window: polygonal boundary
## enclosing rectangle: [-6.40267, 69.97237] x [-7.06032, 100.8694] units
```

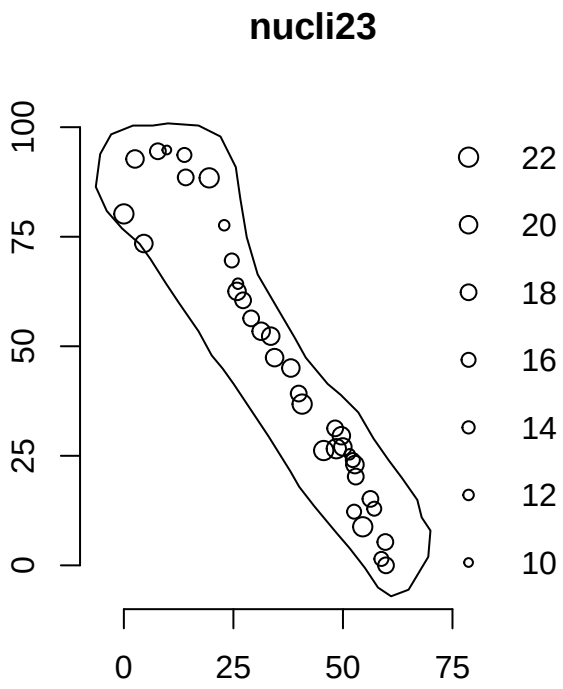
1.4) Briefly describe the marked point process.

```
summary(n23T)
```

```
## Marked planar point pattern:  36 points
## Average intensity 0.01451131 points per square unit
##
## Coordinates are given to 6 decimal places
##
## marks are numeric, of type 'integer'
## Summary:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.00   16.00   18.00   18.17   20.00   22.00
##
## Window: polygonal boundary
## single connected closed polygon with 47 vertices
## enclosing rectangle: [-6.40267, 69.97237] x [-7.06032, 100.8694] units
##                      (76.38 x 107.9 units)
```

```
## Window area = 2480.82 square units
## Fraction of frame area: 0.301
```

```
plot(n23T,main="nucli23",markscale=0.2,leg.side="right")
axis(1,at=c(seq(0,75,by=25)),pos=c(-10,0))
axis(2,at=c(seq(0,100,by=25)),pos=c(-10,-0))
```




```
# Hard to interpret, categorize time into categories for every 4 days instead.

DPOScat=cut(nucli.23$data_pos,breaks=c(9, 12, 14,16, 18, 20, 23),labels=c("10-12","12-14","14-16","16-18
table(DPOScat) # Are no points 12-14. Thus, merge 10-12 and 12-14 into 10-14.
```

```
## DPOScat
## 10-12 12-14 14-16 16-18 18-20 20-22
##     4     0     6    10    10     6
```

```
DPOScat=cut(nucli.23$data_pos,breaks=c(9, 14,16, 18, 20, 22),labels=c("10-14","14-16","16-18", "18-20",
table(DPOScat) # Are no points 12-14. Thus, merge 10-12 and 12-14 into 10-14.
```

```
## DPOScat
## 10-14 14-16 16-18 18-20 20-22
##     4     6    10    10     6
```

```
(n23Tcat=ppp(nucli.23$X,nucli.23$Y,poly=pol.illa,marks=DPOScat))
```

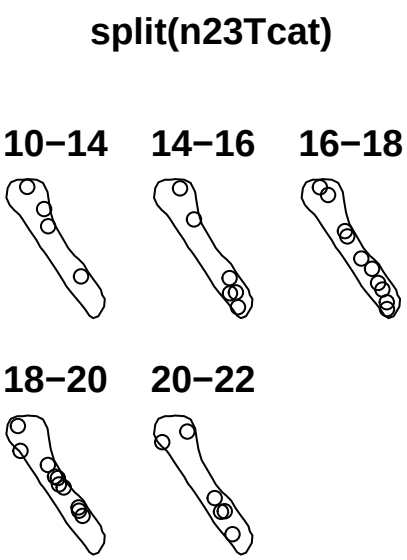
```
## Marked planar point pattern: 36 points
## Multitype, with levels = 10-14, 14-16, 16-18, 18-20, 20-22
## window: polygonal boundary
## enclosing rectangle: [-6.40267, 69.97237] x [-7.06032, 100.8694] units
```

```
summary(n23Tcat)
```

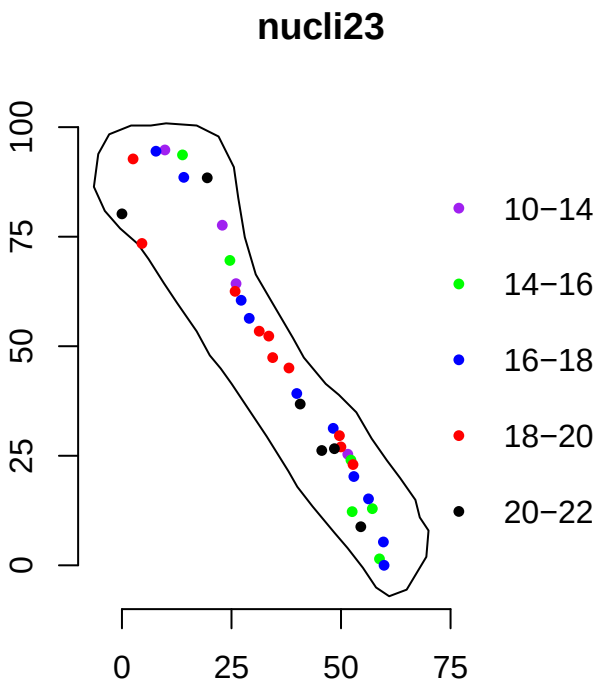
```
## Marked planar point pattern:  36 points
## Average intensity 0.01451131 points per square unit
##
## Coordinates are given to 6 decimal places
```

```
##
## Multitype:
##       frequency proportion   intensity
## 10-14         4  0.1111111 0.001612368
## 14-16         6  0.1666667 0.002418552
## 16-18        10  0.2777778 0.004030920
## 18-20        10  0.2777778 0.004030920
## 20-22         6  0.1666667 0.002418552
##
## Window: polygonal boundary
## single connected closed polygon with 47 vertices
## enclosing rectangle: [-6.40267, 69.97237] x [-7.06032, 100.8694] units
##                      (76.38 x 107.9 units)
## Window area = 2480.82 square units
## Fraction of frame area: 0.301
```

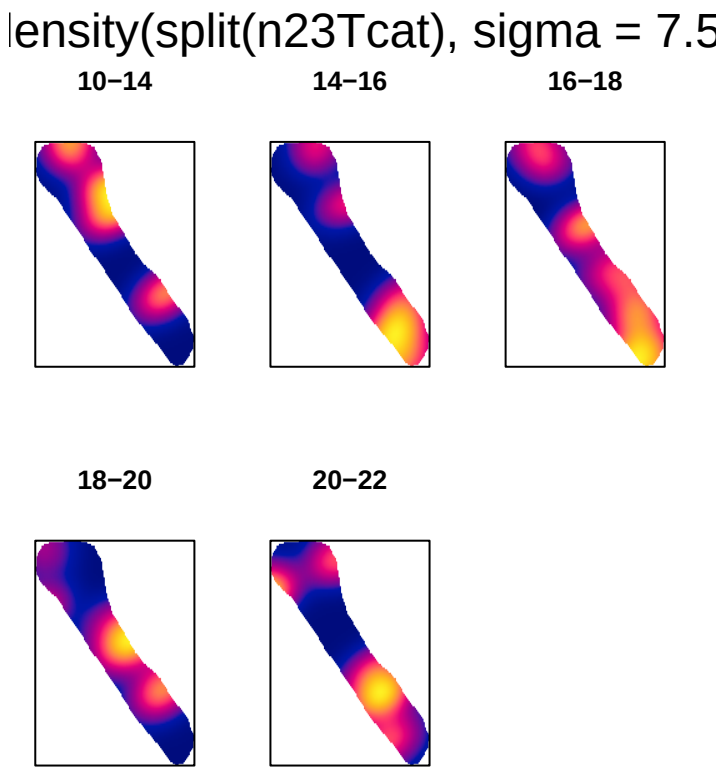
```
plot(split(n23Tcat))
```




```
plot(n23Tcat,main="nucli23",cex=0.75,cols=c("purple", "green", "blue","red", "black"),
     chars=rep(16,4),leg.side="right")
axis(1,at=c(seq(0,75,by=25)),pos=c(-10,0))
axis(2,at=c(seq(0,100,by=25)),pos=c(-10,-0))
```

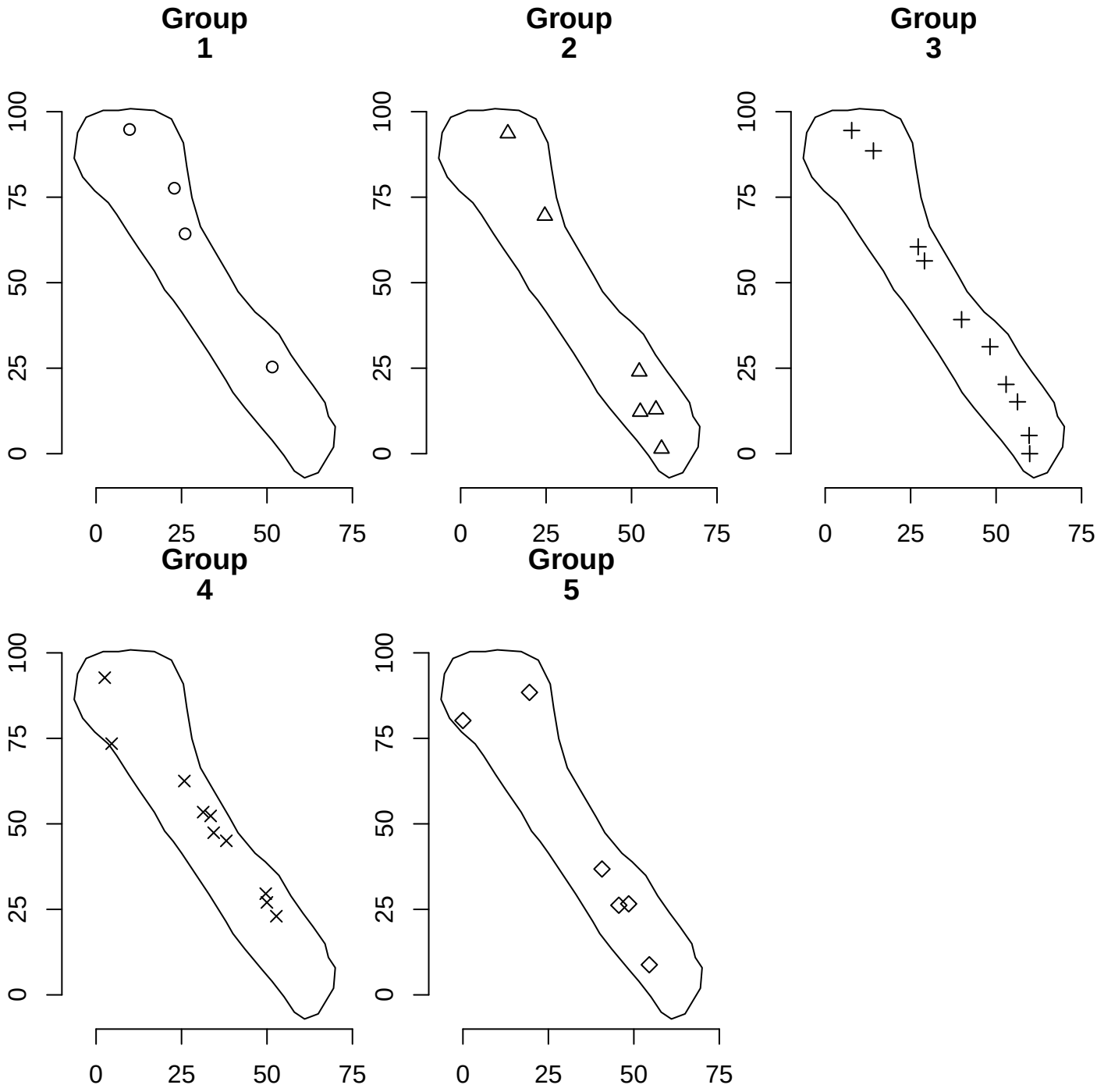



```
plot(density(split(n23Tcat), sigma = 7.5), ribbon = F)
```




```
nucli.23.df=data.frame(nucli.23,DPOScat)
n23new=split(nucli.23.df,DPOScat)
for (i in 1:5){
  temp.ppp=ppp(n23new[[i]]$X,n23new[[i]]$Y,poly=pol.illa)
  plot(temp.ppp,pch=i,main=c("Group",i))
  axis(1,at=c(seq(0,75,by=25)),pos=c(-10,0))
```

```
  axis(2,at=c(seq(0,100,by=25)),pos=c(-10,-0))
  par(ask=T)
}
```




```
par(ask=F)
```

It is hard to see a clear pattern in how the point process evolves with time, but we can make some remarks: It looks like the area we noted as high intensity at first ((x,y) = (60, 25)) stays high intensity through all times that were recorded. One can think that it never gets saturated with nests, i.e. it is a good area for nesting, with room,throughout all recorded time. The other area we noted as relatively high intensity ((x,y)

= (30,60)) is most "popular" for nesting between weeks 16 and 20. There is also an area around (x,y) = (10,90) (in the north-west) that has some nests in all weeks.

Task 2

2.1) Give a point estimate and a 95% confidence interval of the intensity, assuming a homogeneous Poisson process.

When assuming a homogeneous Poisson process, a point estimate of the intensity is simply the average intensity in the window. This means that the point estimate can be read from the summary of the ppp-object,

```
summary(n23)
```

```
## Planar point pattern:  36 points
## Average intensity 0.01451131 points per square unit
##
## Coordinates are given to 6 decimal places
##
## Window: polygonal boundary
## single connected closed polygon with 47 vertices
## enclosing rectangle: [-6.40267, 69.97237] x [-7.06032, 100.8694] units
##                      (76.38 x 107.9 units)
## Window area = 2480.82 square units
## Fraction of frame area: 0.301
```

i.e. the point estimation is 0.0145113. The confidence interval (on the log-scale) can be found from the results shown below

```
model1=ppm(n23, ~1)
model1
```

```
## Stationary Poisson process
## Intensity: 0.01451131
##              Estimate      S.E.   CI95.lo   CI95.hi Ztest      Zval
## log(lambda) -4.232827 0.1666667 -4.559487 -3.906166   *** -25.39696
```

Thus, the 95% confidence interval is (0.0104674, 0.0201175).

2.2) Assess the Completely Spatial Randomness hypothesis.

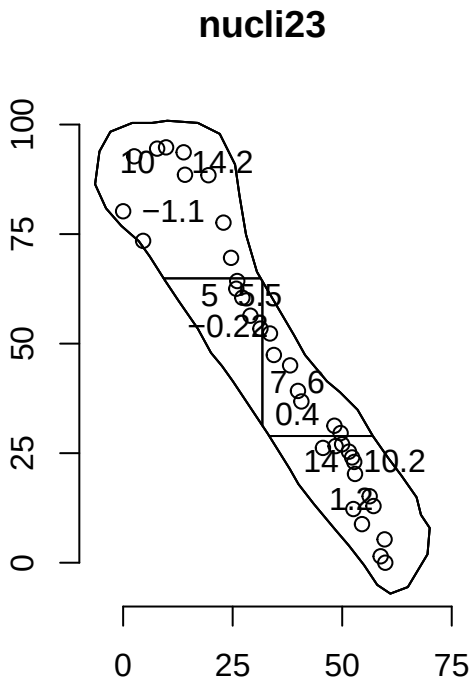
The Completely Spatial Randomness (CSR) hypothesis can be assessed with a chi-squared test based on quadrat counts or a Kolmogorov-Smirnov test. First, let us perform the chi-squared test

```
M <- quadrat.test(n23, nx = 2, ny = 3)
M
```

```
##
##  Chi-squared test of CSR using quadrat counts
##
## data:  n23
## X2 = 2.8589, df = 3, p-value = 0.8278
## alternative hypothesis: two.sided
##
## Quadrats: 4 tiles (irregular windows)
```

```
plot(n23,main="nucli23")
plot(M, add = TRUE)
```

```
axis(1,at=c(seq(0,75,by=25)),pos=c(-10,0))
axis(2,at=c(seq(0,100,by=25)),pos=c(-10,-0))
```



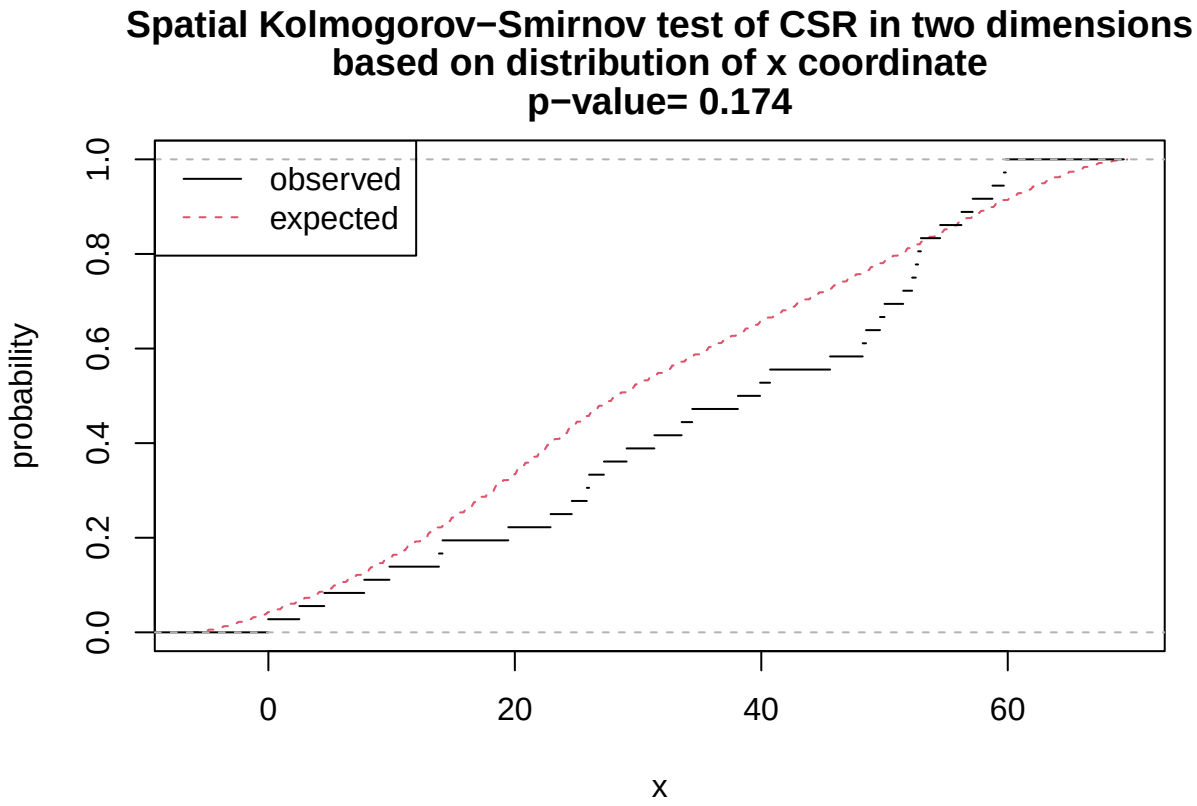
Since the test above has a very large p-value, we cannot conclude that this is NOT a completely random spatial process, since we cannot reject the null hypothesis of CSR. This means that we would conclude that the data is not significantly different from a CSR process.

Next, the Kolmogorov-Smirnov test is performed.

```
KS=cdf.test(n23,covariate="x")
KS
```

```
##
##  Spatial Kolmogorov-Smirnov test of CSR in two dimensions
##
## data:  covariate 'x' evaluated at points of 'n23'
##      and transformed to uniform distribution under CSR
## D = 0.17944, p-value = 0.174
## alternative hypothesis: two-sided
```

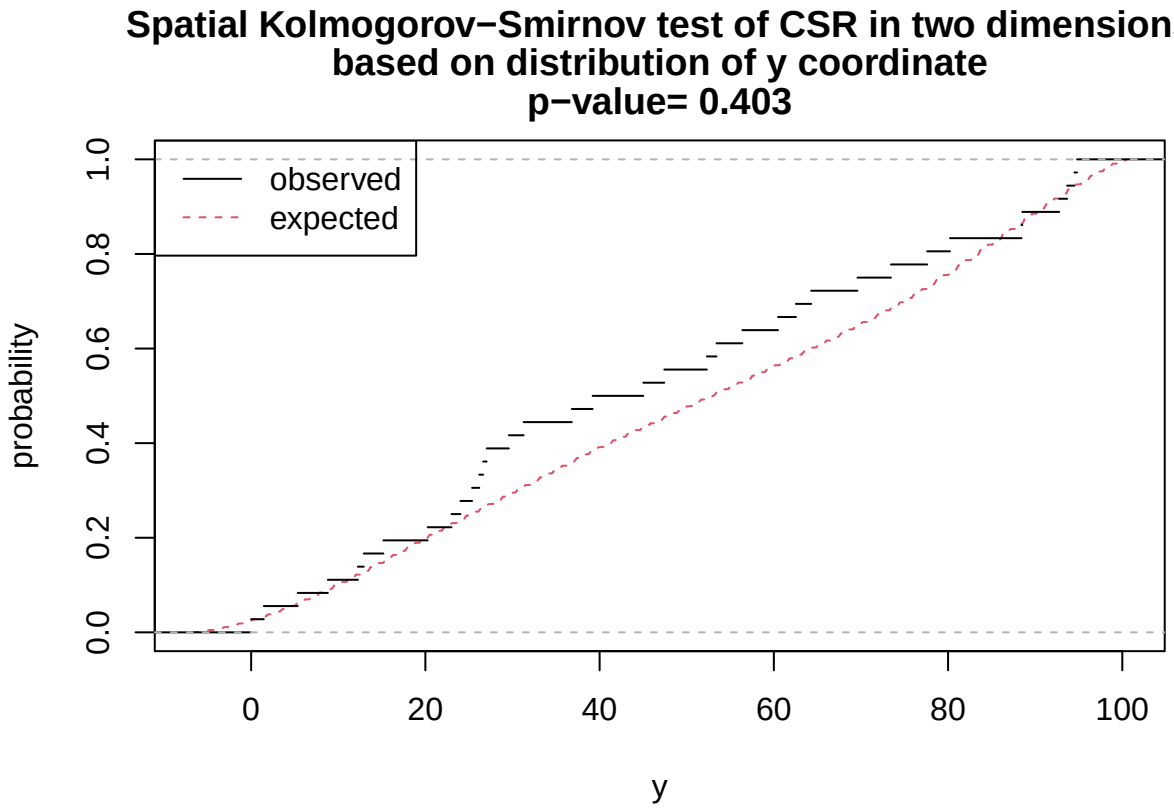
```
plot(KS)
```




```
KS=cdf.test(n23,covariate="y")
KS
```

```
##
##  Spatial Kolmogorov-Smirnov test of CSR in two dimensions
##
## data:  covariate 'y' evaluated at points of 'n23'
##      and transformed to uniform distribution under CSR
## D = 0.14433, p-value = 0.403
## alternative hypothesis: two-sided
```

```
plot(KS)
```



The p-values from this test are also quite large, which means that the Kolmogorov-Smirnov test does not give enough evidence against the null hypothesis of CSR.

Thus, both these two tests lead to the same conclusion; we cannot reject the null-hypothesis of CSR, i.e. we have no evidence of this process not being CSR.

2.3) Fit an inhomogeneous Poisson model to data.

```
# fit n homogeneous poisson model
model1=ppm(n23, ~1)
model1
```

```
## Stationary Poisson process
## Intensity: 0.01451131
##              Estimate      S.E.    CI95.lo    CI95.hi Ztest      Zval
## log(lambda) -4.232827 0.1666667 -4.559487 -3.906166   *** -25.39696
```

```
plot(model1)
```

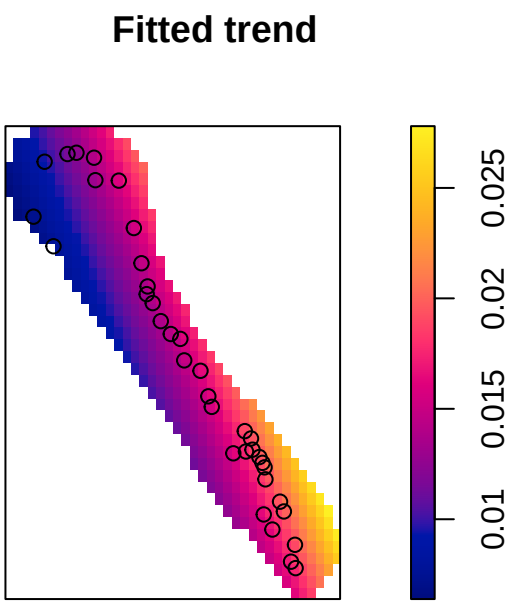
```
## Nothing plotted -- all plots selected are flat surfaces.
```

```
# inhomogeneous Poisson model
model2=ppm(n23, ~x + y)
model2
```

```
## Nonstationary Poisson process
##
## Log intensity:  ~x + y
##
## Fitted trend coefficients:
## (Intercept)           x           y
```

```
## -6.09288023  0.03457419  0.01440722
##
##                Estimate       S.E.     CI95.lo     CI95.hi Ztest       Zval
## (Intercept) -6.09288023 1.49066696 -9.01453379 -3.17122667   *** -4.0873518
## x            0.03457419 0.02315571 -0.01081017  0.07995855        1.4931170
## y            0.01440722 0.01462969 -0.01426646  0.04308089        0.9847927
```

```
plot(model2, se=F)
```

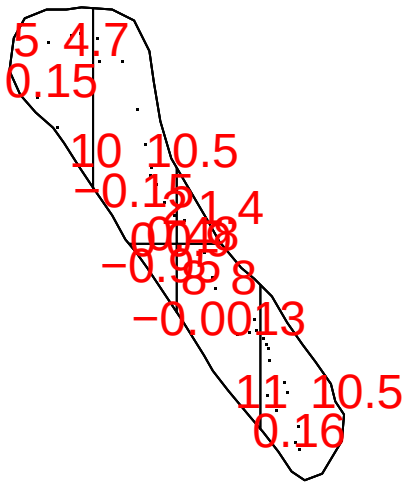


As for the model validation, we have to look at the goodness-of-fit and the residuals.

```
#KS goodness-of-fit
M <- quadrat.test(model2, nx = 4, ny = 2)
M
```

```
##
##  Chi-squared test of fitted Poisson model 'model2' using quadrat counts
##
## data:  data from model2
## X2 = 1.2039, df = 3, p-value = 0.4958
## alternative hypothesis: two.sided
##
## Quadrats: 6 tiles (irregular windows)
```

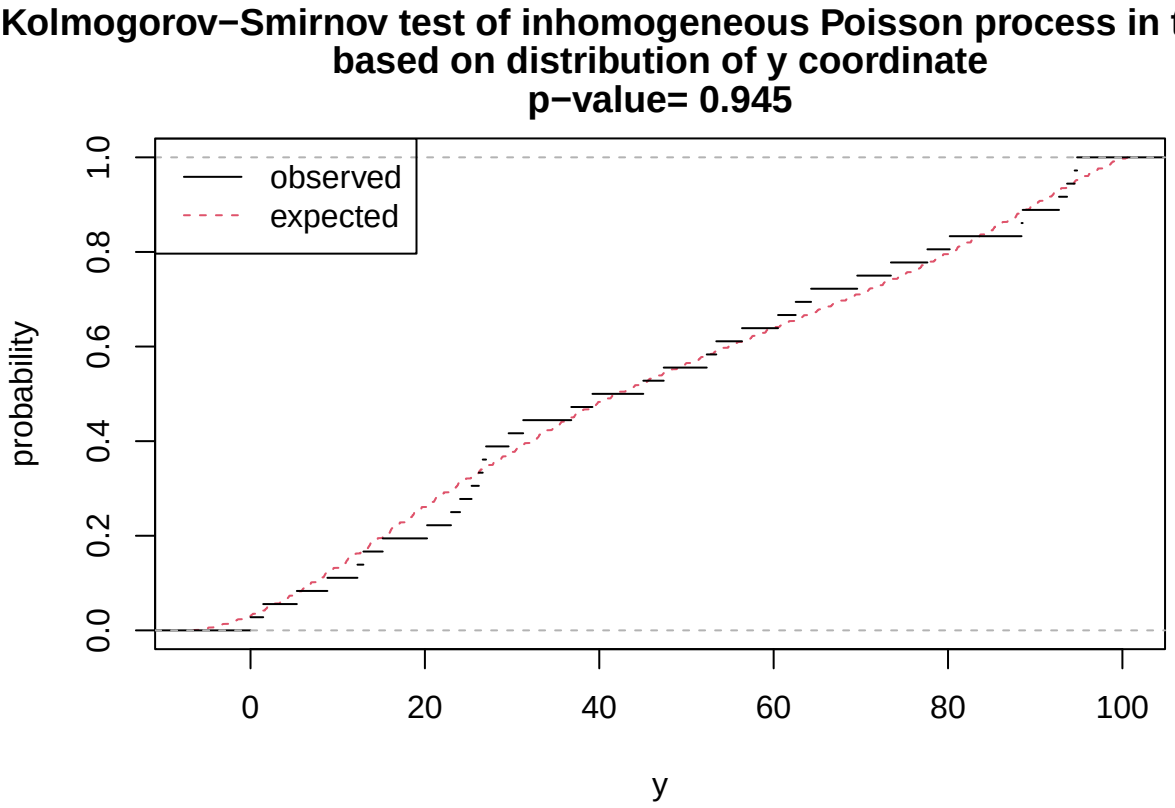
```
plot(n23, pch = ".")
plot(M, add = TRUE, cex = 1.5, col = "red")
```




```
KS=cdf.test(model2,"y")
KS
```

```
##
##  Spatial Kolmogorov-Smirnov test of inhomogeneous Poisson process in
##  two dimensions
##
## data:  covariate 'y' evaluated at points of 'n23'
##      and transformed to uniform distribution under 'model2'
## D = 0.083565, p-value = 0.945
## alternative hypothesis: two-sided
```

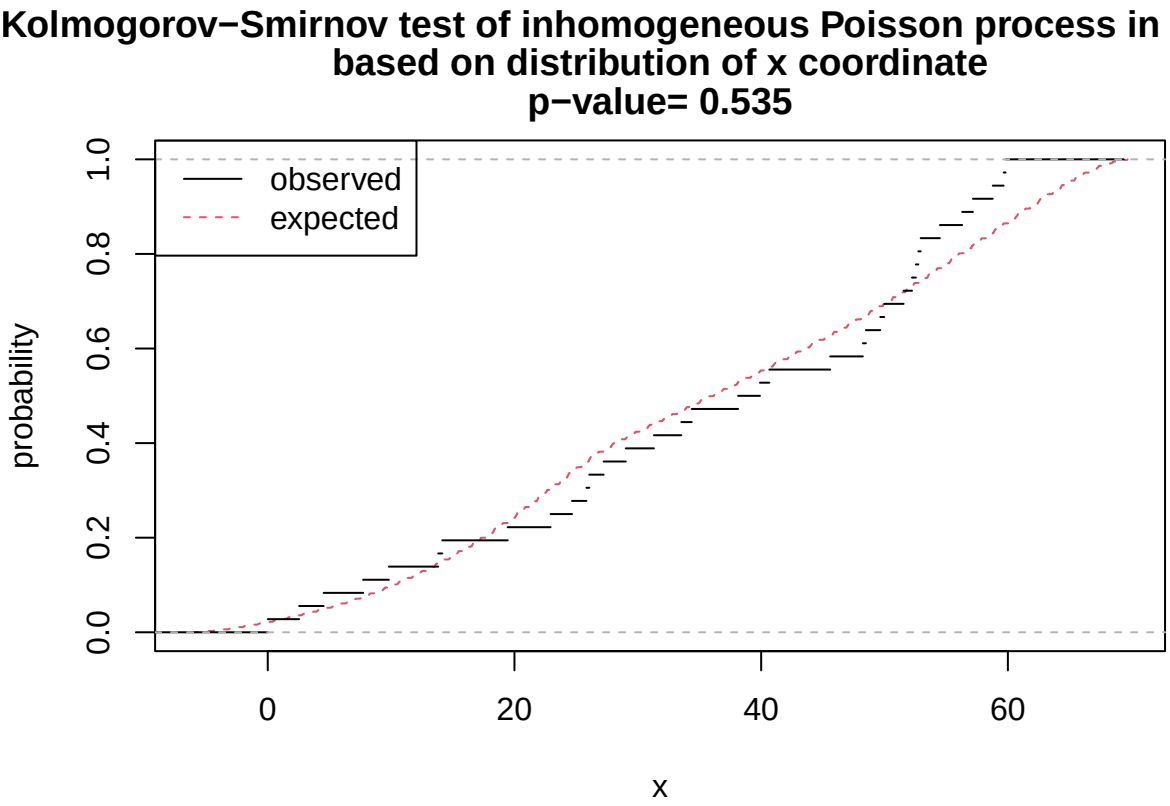
```
plot(KS)
```




```
KS=cdf.test(model2,"x")
KS
```

```
##
##  Spatial Kolmogorov-Smirnov test of inhomogeneous Poisson process in
##  two dimensions
##
## data:  covariate 'x' evaluated at points of 'n23'
##      and transformed to uniform distribution under 'model2'
## D = 0.12992, p-value = 0.535
## alternative hypothesis: two-sided
```

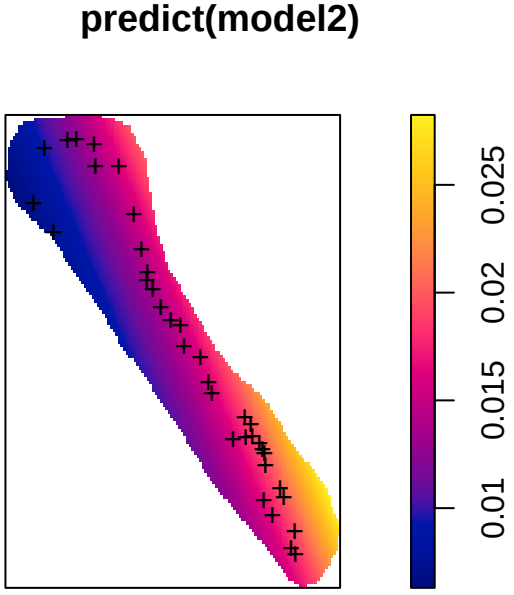
```
plot(KS)
```



As we can see form the Kolmogorov-Smirnov test, the observed values are very similar to those expected. Besides the p-value is 0.2339 therefore we can not reject the null hypothesis meaning that the model is true.

In order to go further the validation with residuals is done.

```
#Validation with residuals
plot(predict(model2))
plot(n23, add = TRUE, pch = "+")
```




```
sum(eem(model2))/area(n23)
```

```
## [1] 0.9843835
```

We see that the prediction is quite accurate with the plot of the points. Besides the parameter of the fitting is near 1 meaning that this models fits well as a Poisson. Therefore the inhomogeneous Poisson model is a good model for describing this situation of nesting.

Task 3

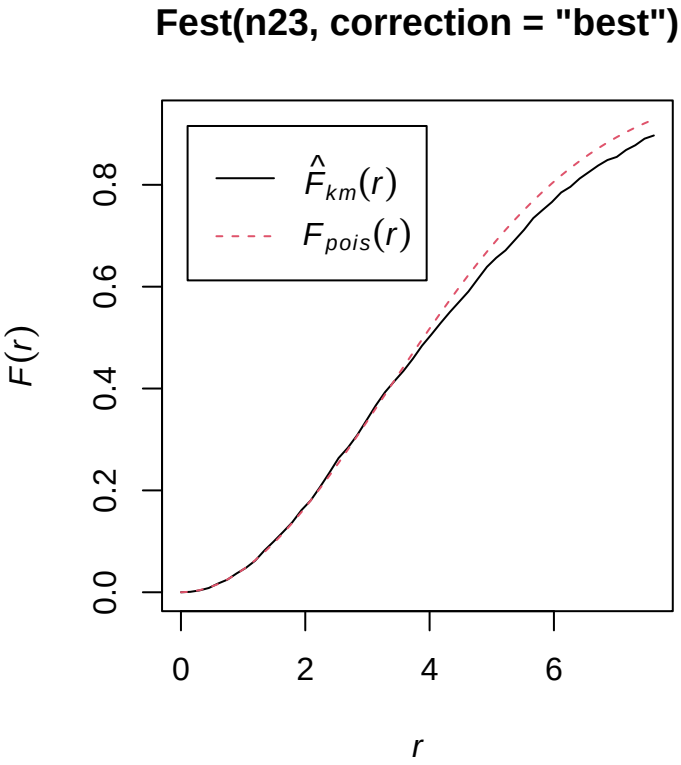
3.1) Explore the pattern of interaction.

First we see the Empty Space distances in order to see the distance of a fixed location in the window to the nearest data point.

```
# Empty space function
Fc <- Fest(n23)
Fc
```

```
## Function value object (class 'fv')
## for the function r -> F(r)
## ....................................................................
##         Math.label      Description
## r       r               distance argument r
## theo    F[pois](r)      theoretical Poisson F(r)
## cs      hat(F)[cs](r)   Chiu-Stoyan estimate of F(r)
## rs      hat(F)[bord](r) border corrected estimate of F(r)
## km      hat(F)[km](r)   Kaplan-Meier estimate of F(r)
## hazard  hat(h)[km](r)   Kaplan-Meier estimate of hazard function h(r)
## theohaz h[pois](r)      theoretical Poisson hazard h(r)
## ....................................................................
## Default plot formula:  .~r
## where "." stands for 'km', 'rs', 'cs', 'theo'
## Recommended range of argument r: [0, 7.6077]
## Available range of argument r: [0, 15.961]
```

```
par(pty = "s")
plot(Fest(n23,correction="best"),legend=T)
```



We can see how the Edge corrected distribution function (the black line) is a little bit below of the Homogeneous Poisson Distribution (red line). This shows we have a Clustered Pattern in this case.

If we see the envelope for the pattern interaction:

```
E <- envelope(n23, Fest, nsim = 19, rank = 1, global = TRUE,correction="best")
```

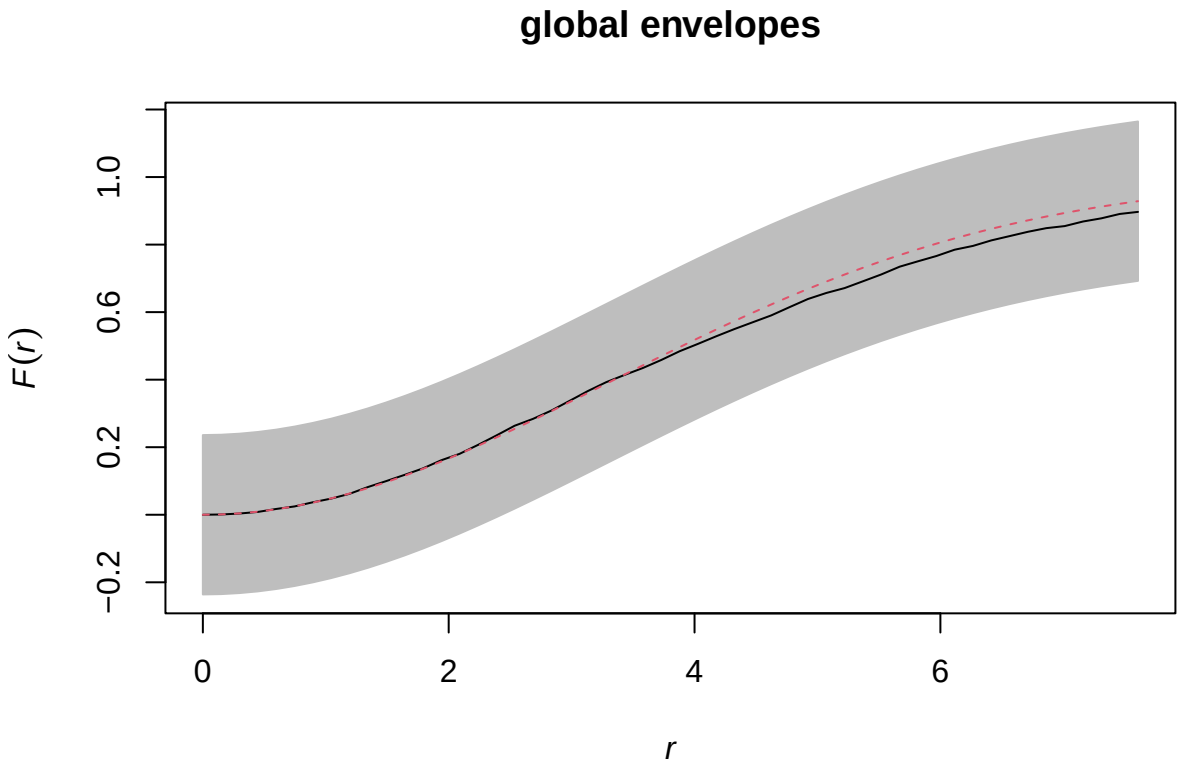
```
## Generating 19 simulations of CSR  ...
## 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,  19.
##
## Done.
```

```
E
```

```
## Simultaneous critical envelopes for F(r)
## and observed value for 'n23'
## Edge correction: "km"
## Obtained from 19 simulations of CSR
## Envelope based on maximum deviation of F(r) from null value for CSR (known
## exactly)
## Alternative: two.sided
## Significance level of simultaneous Monte Carlo test: 1/20 = 0.05
## ------------------------------------------------------------
##      Math.label     Description
## r    r              distance argument r
## obs  hat(F)[obs](r) observed value of F(r) for data pattern
## theo F[theo](r)     theoretical value of F(r) for CSR
## lo   hat(F)[lo](r)  lower critical boundary for F(r)
## hi   hat(F)[hi](r)  upper critical boundary for F(r)
## ------------------------------------------------------------
## Default plot formula:  .~r
## where "." stands for 'obs', 'theo', 'hi', 'lo'
```

```
## Columns 'lo' and 'hi' will be plotted as shading (by default)
## Recommended range of argument r: [0, 7.6077]
## Available range of argument r: [0, 7.6077]
```

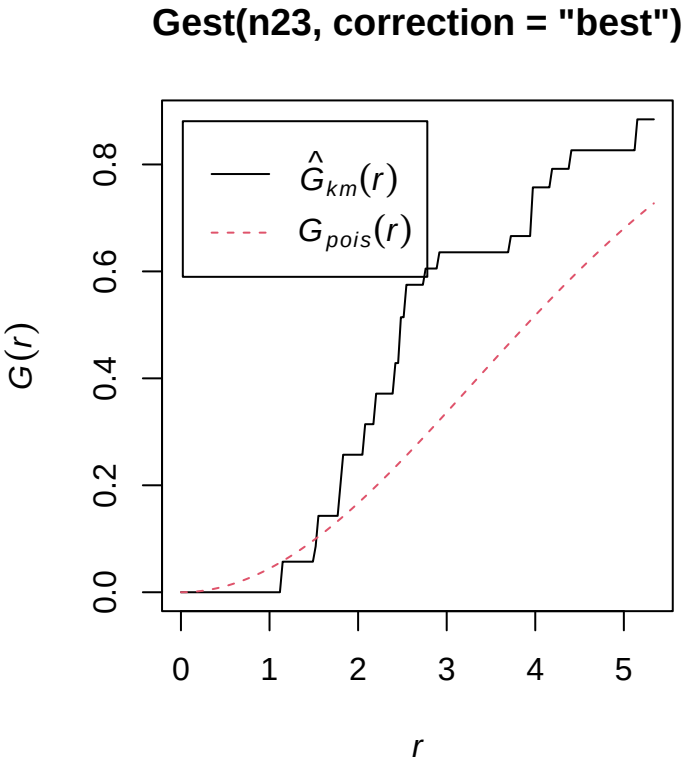
```
plot(E, main = "global envelopes",legend=F)
```



We can see that the data fits well in this envelope and follows the simulated interaction model.

Considering now the Nearest Neighbors distances:

```
# NN function
par(pty = "s")
plot(Gest(n23,correction="best"),legend=T)
```



We can see how the Edge corrected distribution function (the black line) is above of the Homogeneous Poisson Distribution (red line). This shows we have a Clustered Pattern for the Nearest Neighbors distances.

If we see the envelope for the pattern interaction:

```
E <- envelope(n23, Gest, nsim = 19, rank = 1, global = TRUE,correction="best")
```

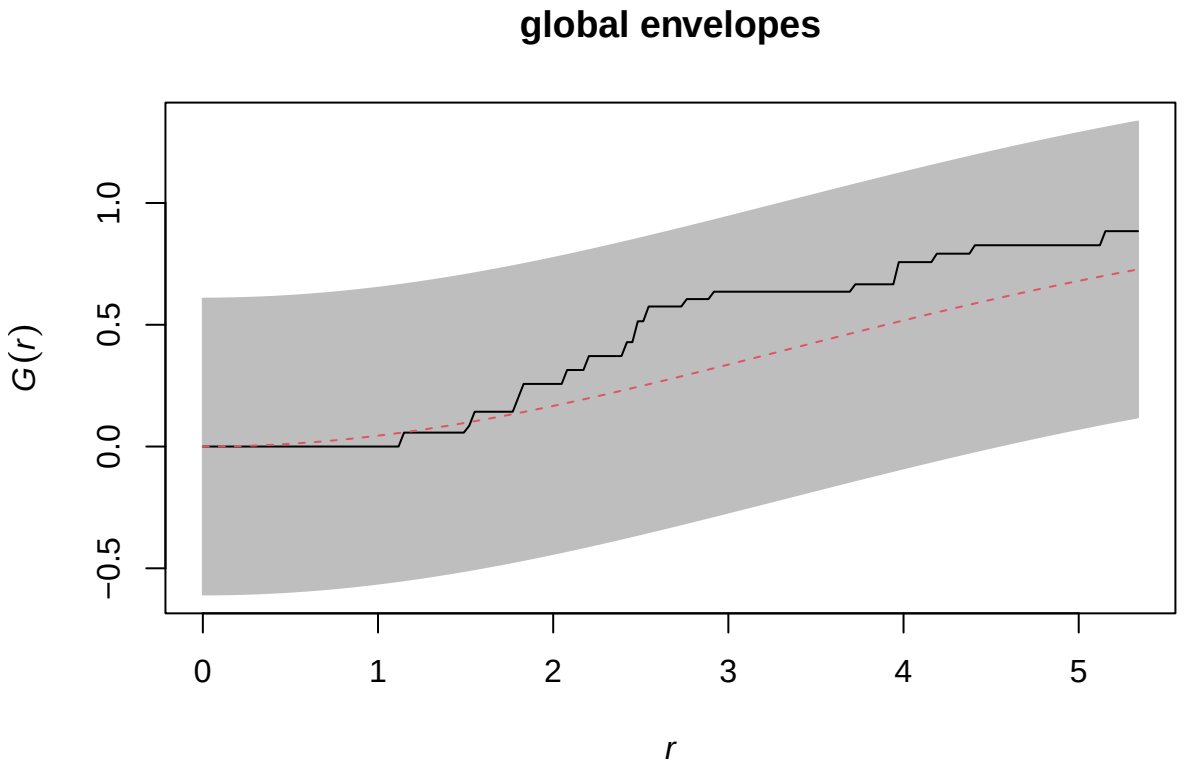
```
## Generating 19 simulations of CSR  ...
## 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,  19.
##
## Done.
```

```
E
```

```
## Simultaneous critical envelopes for G(r)
## and observed value for 'n23'
## Edge correction: "km"
## Obtained from 19 simulations of CSR
## Envelope based on maximum deviation of G(r) from null value for CSR (known
## exactly)
## Alternative: two.sided
## Significance level of simultaneous Monte Carlo test: 1/20 = 0.05
## ------------------------------------------------------------
##      Math.label     Description
## r    r              distance argument r
## obs  hat(G)[obs](r) observed value of G(r) for data pattern
## theo G[theo](r)     theoretical value of G(r) for CSR
## lo   hat(G)[lo](r)  lower critical boundary for G(r)
## hi   hat(G)[hi](r)  upper critical boundary for G(r)
## ------------------------------------------------------------
## Default plot formula:  .~r
## where "." stands for 'obs', 'theo', 'hi', 'lo'
```

```
## Columns 'lo' and 'hi' will be plotted as shading (by default)
## Recommended range of argument r: [0, 5.3385]
## Available range of argument r: [0, 5.3385]
```

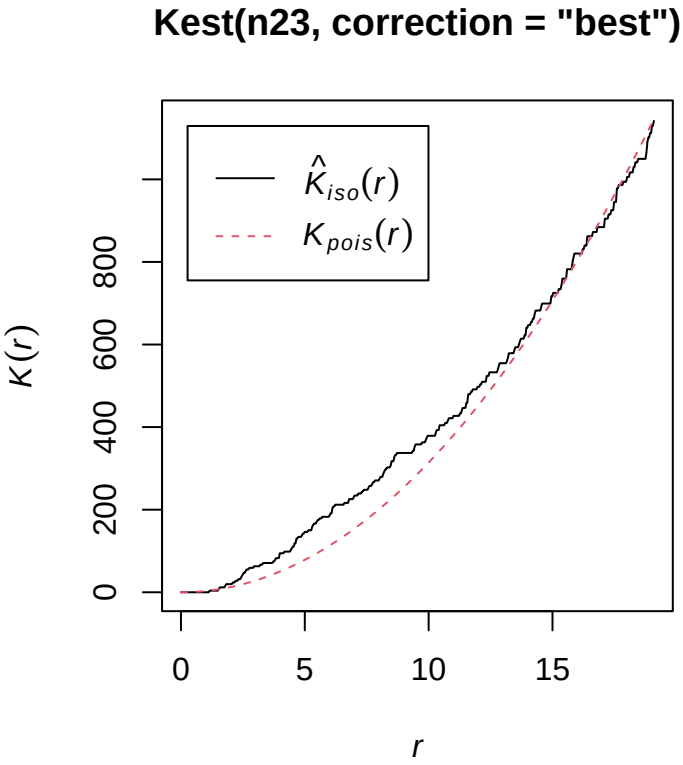
```
plot(E, main = "global envelopes",legend=F)
```



Again we can see that the data fits well in this envelope and follows the simulated interaction model.

Finally we consider the Pairwise distances

```
# Pairwise distances
par(pty = "s")
plot(Kest(n23,correction="best"),legend=T)
```



We can see how the Edge corrected distribution function (the black line) is above of the Homogeneous Poisson Distribution (red line). This shows we have a Clustered Pattern for the Pairwise distances.

If we see the envelope for the pattern interaction:

```
E <- envelope(n23, Kest, nsim = 19, rank = 1, global = TRUE,correction="best")
```

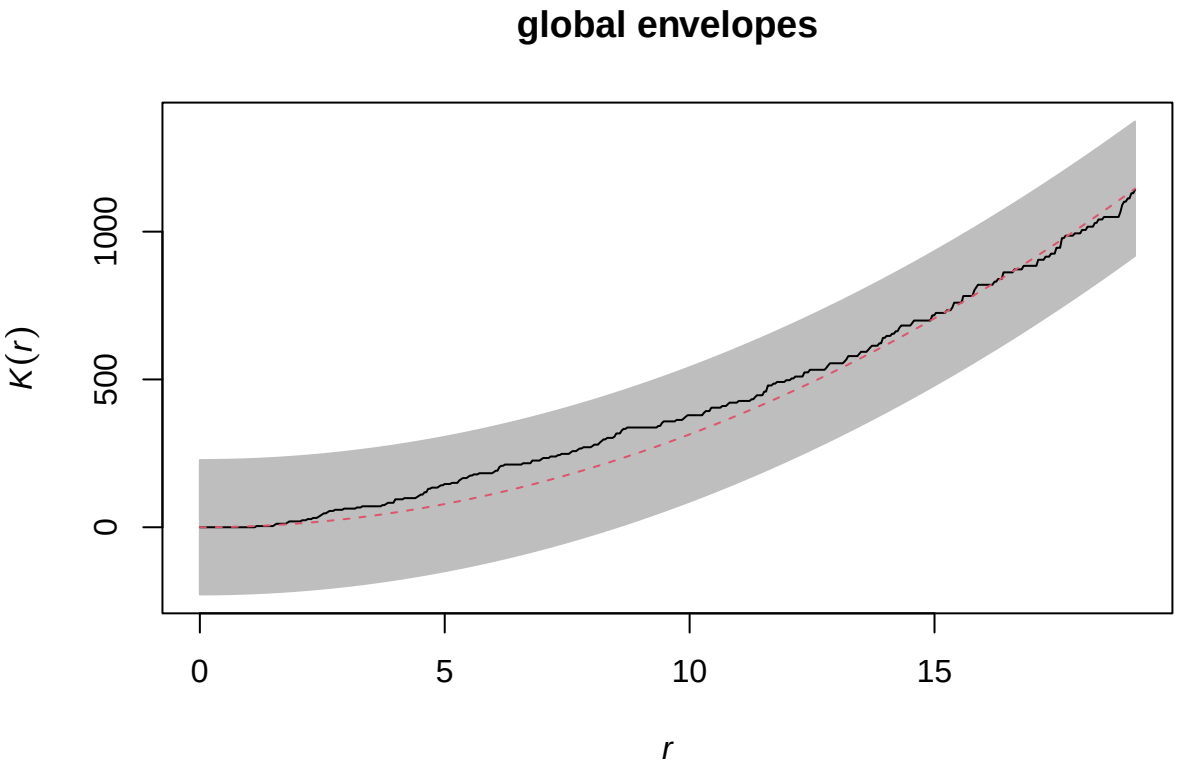
```
## Generating 19 simulations of CSR  ...
## 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,  19.
##
## Done.
```

```
E
```

```
## Simultaneous critical envelopes for K(r)
## and observed value for 'n23'
## Edge correction: "iso"
## Obtained from 19 simulations of CSR
## Envelope based on maximum deviation of K(r) from null value for CSR (known
## exactly)
## Alternative: two.sided
## Significance level of simultaneous Monte Carlo test: 1/20 = 0.05
## ..........................................................
##      Math.label     Description
## r    r              distance argument r
## obs  hat(K)[obs](r) observed value of K(r) for data pattern
## theo K[theo](r)     theoretical value of K(r) for CSR
## lo   hat(K)[lo](r)  lower critical boundary for K(r)
## hi   hat(K)[hi](r)  upper critical boundary for K(r)
## ..........................................................
## Default plot formula:  .~r
## where "." stands for 'obs', 'theo', 'hi', 'lo'
```

```
## Columns 'lo' and 'hi' will be plotted as shading (by default)
## Recommended range of argument r: [0, 19.094]
## Available range of argument r: [0, 19.094]
```

```
plot(E, main = "global envelopes",legend=F)
```

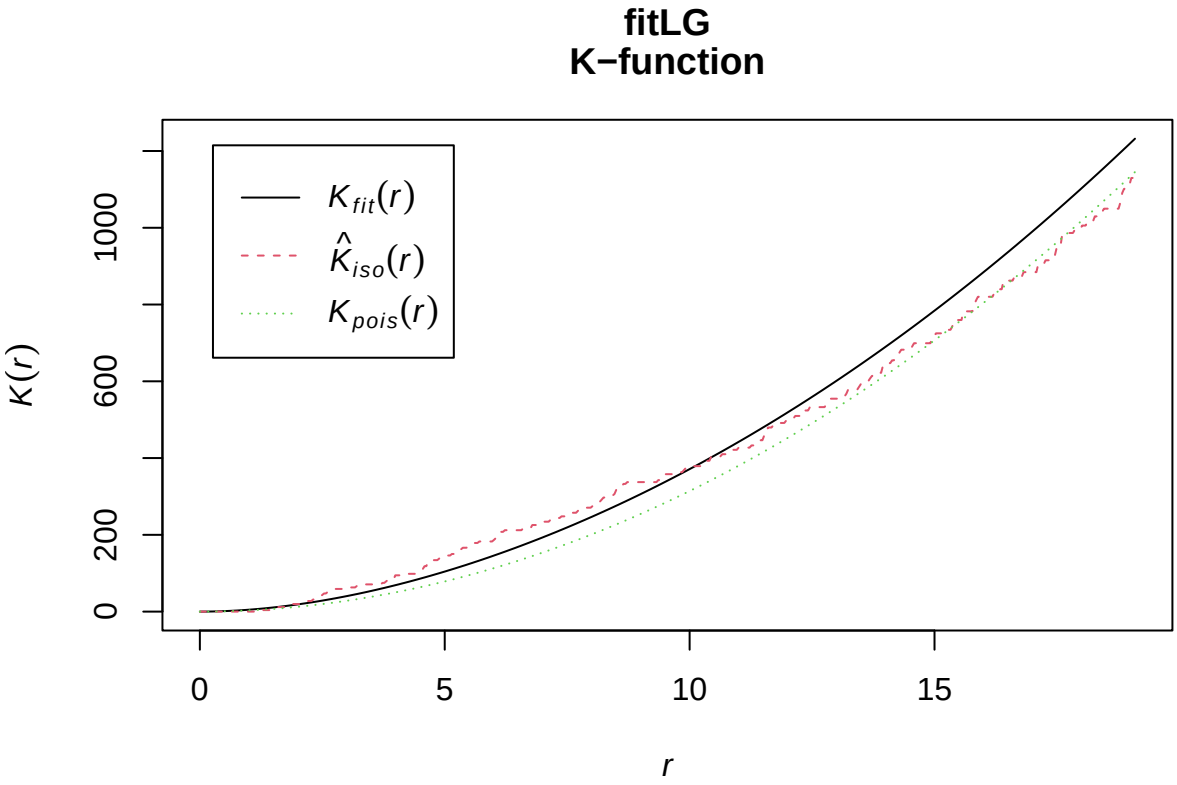


We can see again that the data fits well in this envelope and follows the simulated interaction model.

So we can conclude we have some Clusters and the Poisson model fits well the data.

3.2) Fit a log-Gaussian Cox process to data. Comment the results.

```
fitLG <- kppm(n23, ~1, "LGCP")

plot(fitLG,legend=T,what="statistic")
```



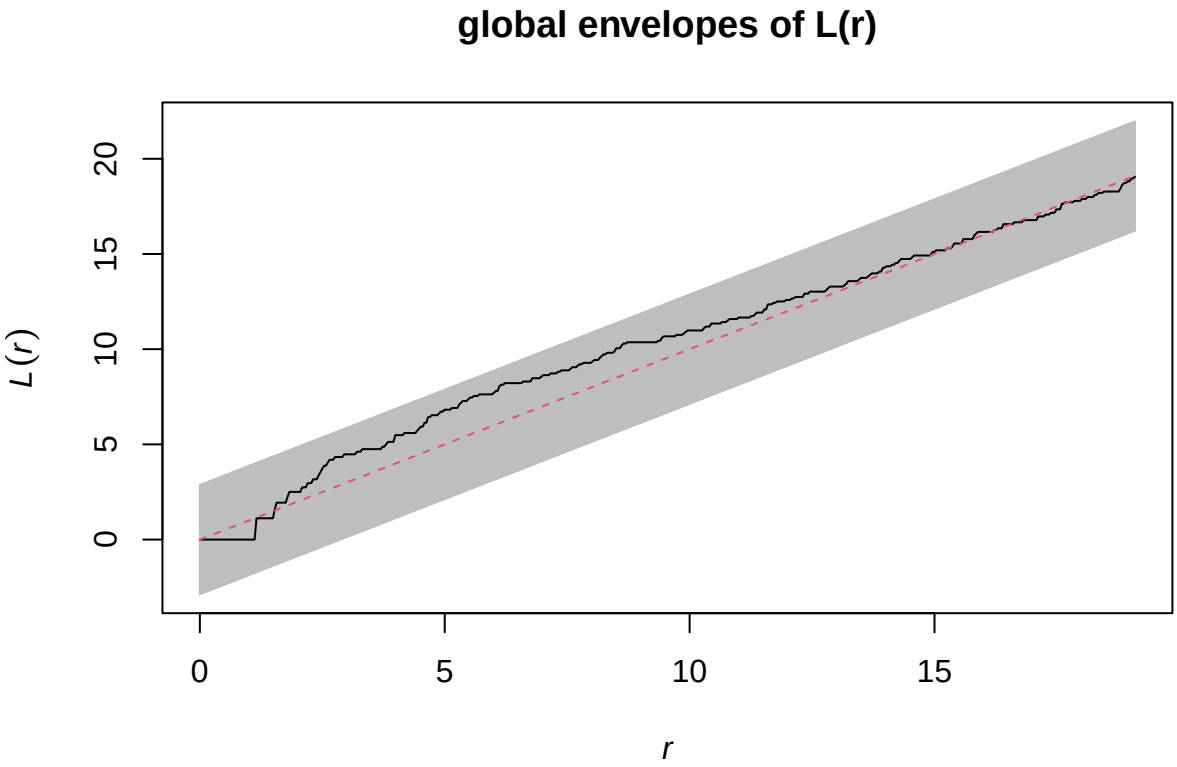
It can be observed that the Cox-Gauss process, represented by the black line fits better to the data (represented with the red line) than the Poisson model (represented with the green line) in short distances. But in long distances the poisson model fits better the data. In the global computation it seems to be better to model the clusters interactions with the Cox-Gauss model.

3.3) Fit a Gibbs process model to data. Comment the results

```
##################
# Gibbs process ##
##################

E <- envelope(n23, Lest, nsim = 19, rank = 1, global = TRUE,correction="best")
```

```
## Generating 19 simulations of CSR  ...
## 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,  19.
##
## Done.
```

```
plot(E, main = "global envelopes of L(r)", legend=F)
```



This graphic show us that the data is well fitted in the Gibbs process (because the black line representing our data interaction is in the gibbs model envoirment represented as the grey area), so our data clusters and regular points interact themselves in a similar way as the Giibs proces model do.

Task 4

Using the Primary Biliary Cirrhosis data (PBC) make a case-control point pattern analysis.

```
## Registered S3 method overwritten by 'cli':
##   method     from
##   print.boxx spatstat.geom
##
## -- Column specification --------------------------------------------------------
## cols(
##   x = col_double(),
##   y = col_double(),
##   marks = col_character()
## )

##
## -- Column specification --------------------------------------------------------
## cols(
##   x = col_double(),
##   y = col_double()
## )
```

Data overview: in this case we have a set of data with the coordinates of the points in which there is the success (biliary cirrhosis) alongside a mark indicating whether it is a case or a control. By comparing both we will be able to see if the case is notably different and therefore it has to be a considerable phenomenon.

And by that if there is something that accentuates or diminishes the prevalence of biliary cirrhosis.

First of all the data has to be transformed into spatial point data. And separate the cases from the controls.

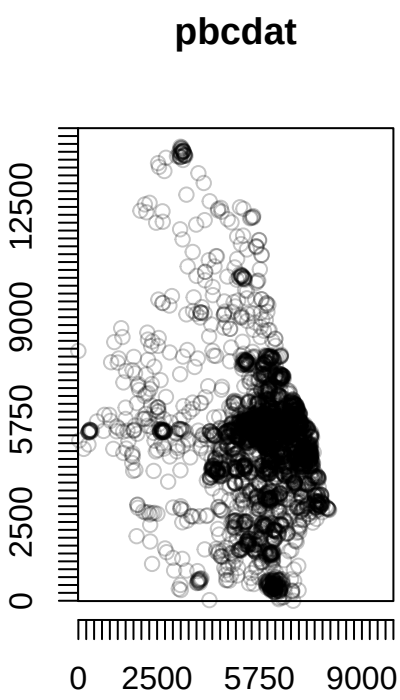
```
pbcdata <- read.table(file="PBCdata.txt", header=T, sep="\t",stringsAsFactors = T)
summary(pbcdata)
```

```
##        x               y             marks
##  Min.   :36670   Min.   :51010   case   : 736
##  1st Qu.:42000   1st Qu.:55040   control:2769
##  Median :42680   Median :56210
##  Mean   :42461   Mean   :55974
##  3rd Qu.:43250   3rd Qu.:56780
##  Max.   :44630   Max.   :65420
```

```
min.X<-min(pbcdata$x)
min.Y<-min(pbcdata$y)

pbcdata$x<-pbcdata$x-min.X
pbcdata$y<-pbcdata$y-min.Y
summary(pbcdata)
```

```
##        x              y             marks
##  Min.   :   0   Min.   :    0   case   : 736
##  1st Qu.:5330   1st Qu.: 4030   control:2769
##  Median :6010   Median : 5200
##  Mean   :5791   Mean   : 4964
##  3rd Qu.:6580   3rd Qu.: 5770
##  Max.   :7960   Max.   :14410
```

```
pbcdat <-ppp(pbcdata$x, pbcdata$y, c(-10, 10000), c(-10, 15000))
plot(pbcdat)
axis(1,at=c(seq(0,10000,by=250)))
axis(2,at=c(seq(0,15000,by=250)),pos=c(-10,0))
```



```
pbcpoly=read.table(file=paste("PBCpoly.txt"),header=T,sep="\t")
summary(pbcpoly)
```

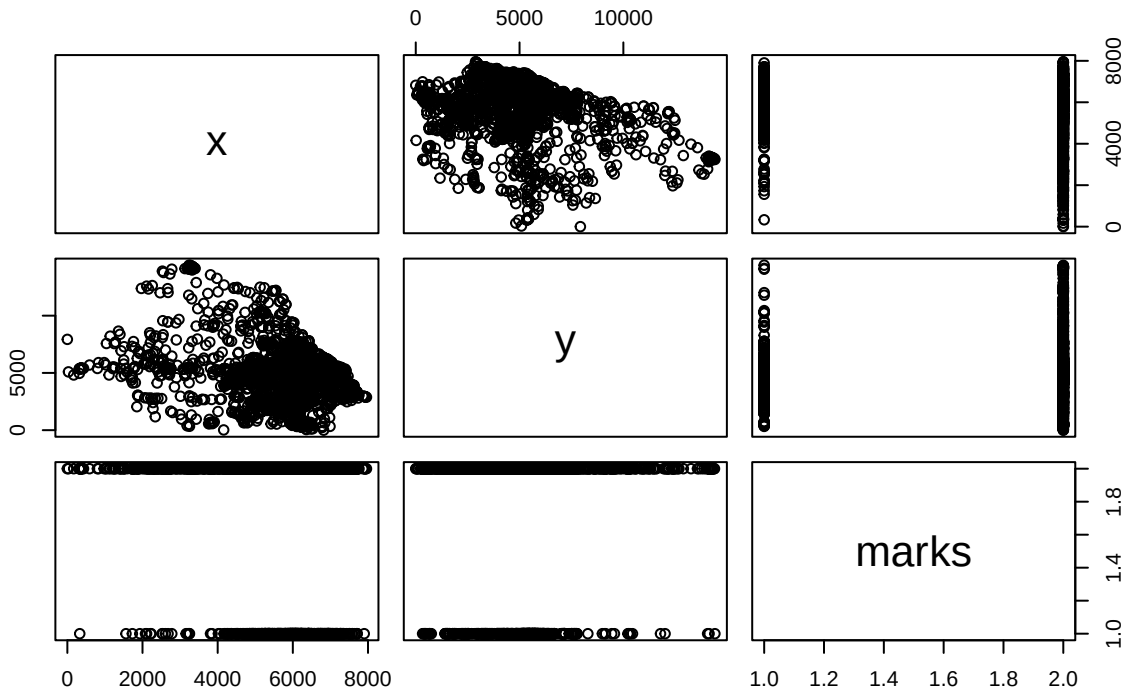
```
##        x               y        
##  Min.   :35603   Min.   :50640  
##  1st Qu.:37716   1st Qu.:52976  
##  Median :40162   Median :56946  
##  Mean   :40221   Mean   :57100  
##  3rd Qu.:43025   3rd Qu.:61012  
##  Max.   :44770   Max.   :66171
```

```
pbcpoly$x <- pbcpoly$x-min.X
pbcpoly$y <- pbcpoly$y-min.Y
summary(pbcpoly)
```

```
##        x               y         
##  Min.   :-1067   Min.   : -370.5  
##  1st Qu.: 1046   1st Qu.: 1965.8  
##  Median : 3492   Median : 5936.3  
##  Mean   : 3551   Mean   : 6090.5  
##  3rd Qu.: 6355   3rd Qu.:10002.2  
##  Max.   : 8100   Max.   :15160.9
```

```
pbcdata$marks<- as.factor(pbcdata$marks)
pol.illa<-list(x=pbcpoly$x,y=pbcpoly$y)
plot(owin(poly=pol.illa),main="plot poly")
```


```
pbcdata_poly=ppp(pbcdata$x,pbcdata$y,poly=pol.illa, marks=pbcdata$marks)
plot(pbcdata,main="pbcdata")
axis(1,at=c(seq(-0,10000,by=250)),pos=c(-500,0))
axis(2,at=c(seq(-100,15000,by=250)),pos=c(-1500,0))
```



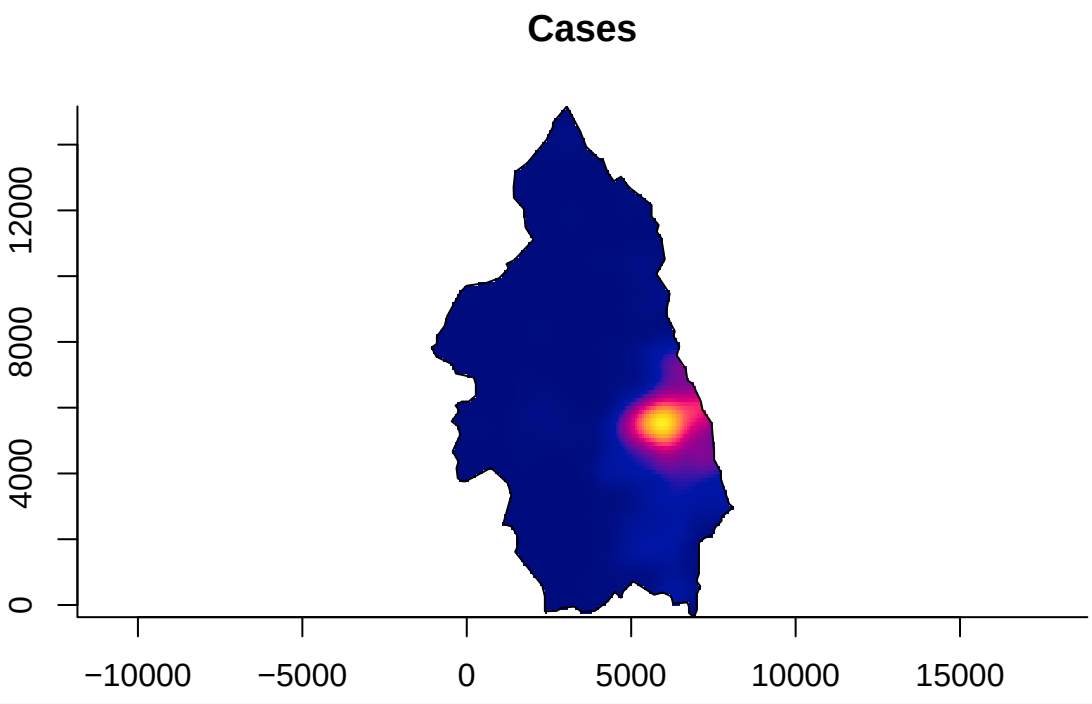

```
cases <- split(pbcdata_poly)$case
control <- split(pbcdata_poly)$control
```

Once the data has been separated and transformed we can compute and plot the relative risk, comparing the cases with the control.

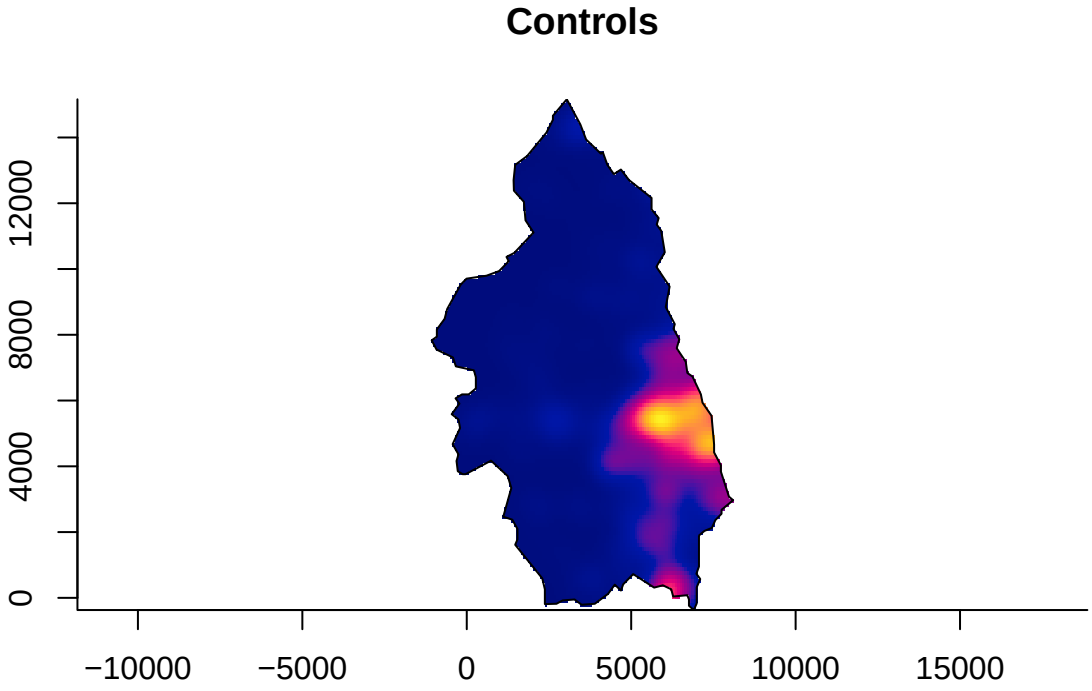
```
#Intensities
pcb <- risk(cases,control, log=F, intensity=T)
```

```
## Estimating case and control densities...Done.
```

```
#Cases
plot(pcb$f, ribbon=F, main="Cases")
```

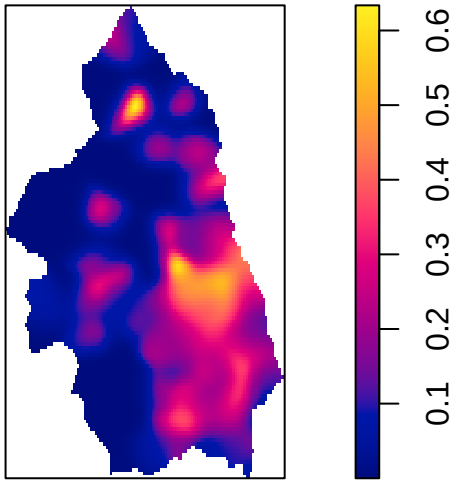



```
#Controls
plot(pcb$g, ribbon=F, main="Controls")
```




```
#Relative risk
plot(pcb$rr, main="Relative risk")
```

Relative risk




```
#Reference value
ref_val <-cases$n/control$n
```

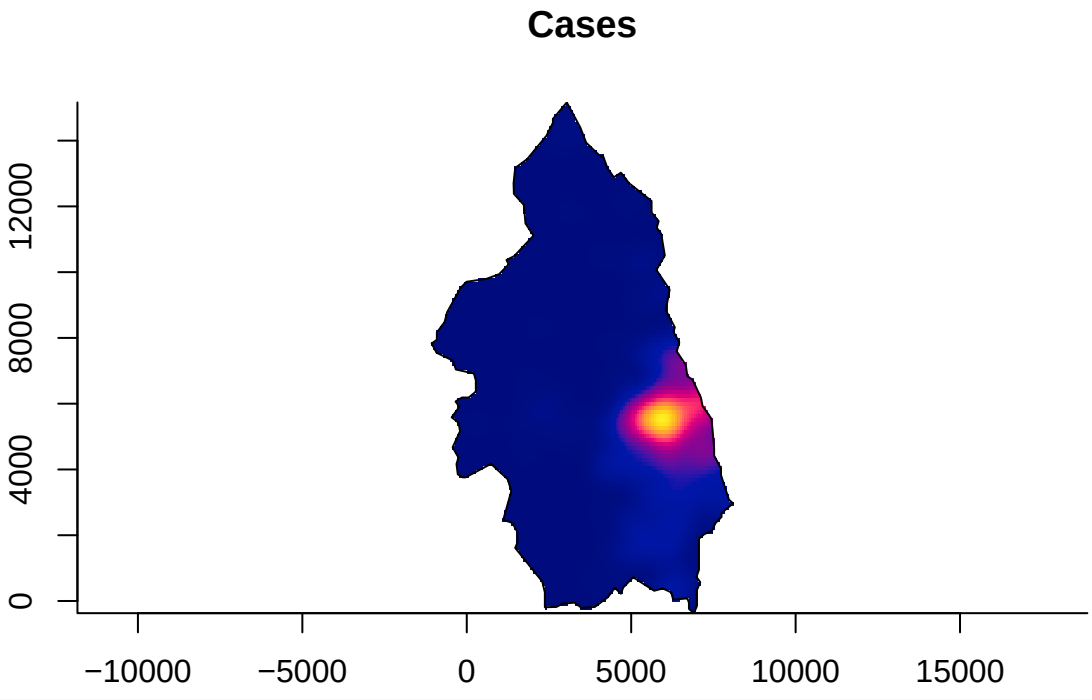
In the first plot, are plotted the cases were we can see there is a high intensity area with some lower density around it. When we compare it with the second plot, the controls, the patter is similar but with greater intensity, When computing the relative risk the graphic changes quite a bit, with the remark of the cluster were previously we saw high risk but with another area of high risk which was not present in the previous graphs. In order to be able to ponderate, the reference value, which is the ratio between the number of cases and controls so we can acknowledge the fact that there will be more general intensity (clear spots) in the control plot. having as reference value `ref_val`

Now let's try if doing the logarithm of the risk we are able to reduce the difference between cases and controls with the relative risk.

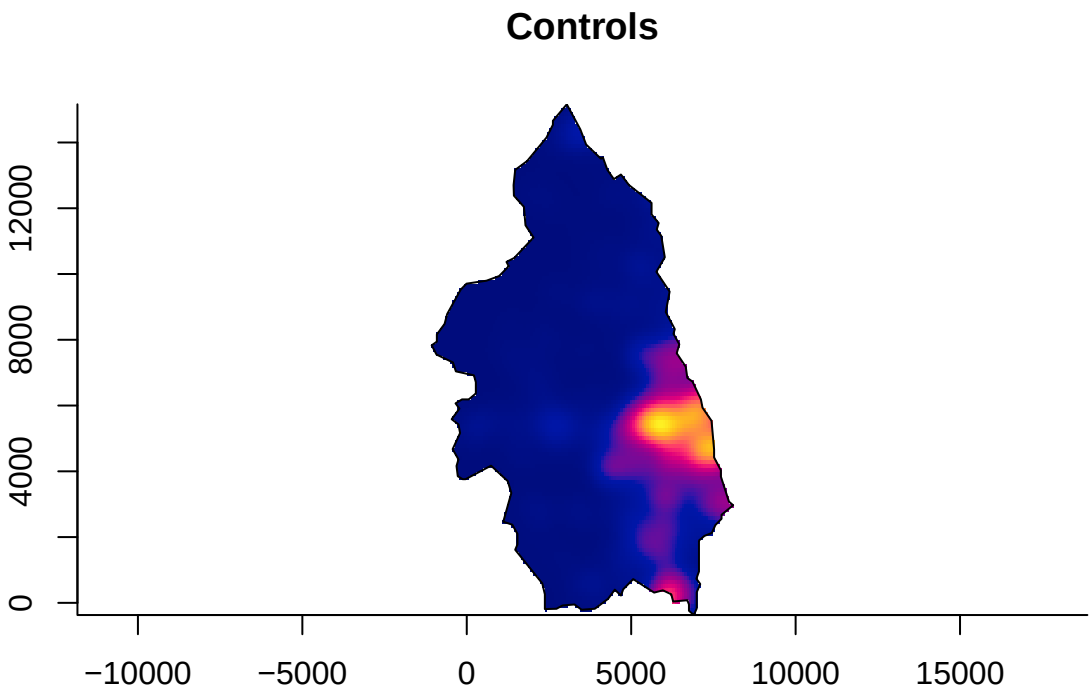
```
#Log intensities
pcb <- risk(cases, control, log=T, intensity=T)
```

```
## Estimating case and control densities...Done.
```

```
#Cases
plot(pcb$f, ribbon=F, main="Cases")
```

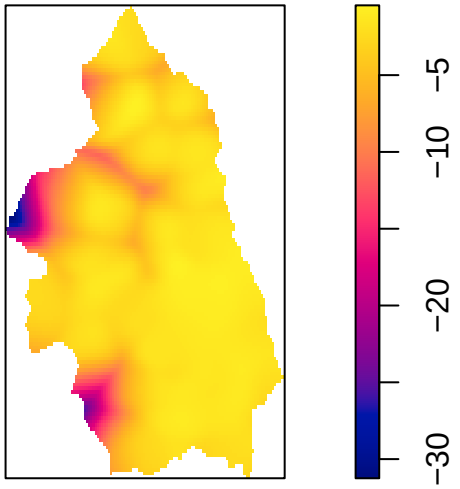



```
#Controls
plot(pcb$g, ribbon=F, main="Controls")
```




```
#Relative risk
plot(pcb$rr, main="Relative risk")
```

Relative risk




```
#Reference value
cases$n/control$n
```

```
## [1] 0.2657999
```

```
#Sealevel -->yellow
max(pcb$rr)
```

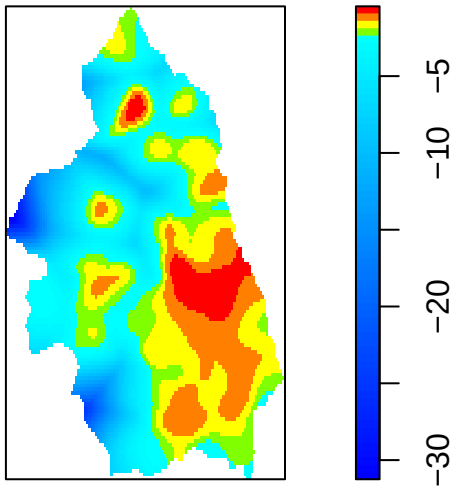
```
## [1] -0.4561877
```

```
min(pcb$rr)
```

```
## [1] -31.24634
```

```
plot(pcb$rr, col=beachcolours(c(-8,-1), sealevel = log(cases$n/control$n)), main="Corrected relative ris
```

Corrected relative risk



With the log transformed risk applied both plots, “Cases” and “Controls” remain the same but the relative risk is changed through an augmentation (or clarification of differences). The problem is that the plot lacks

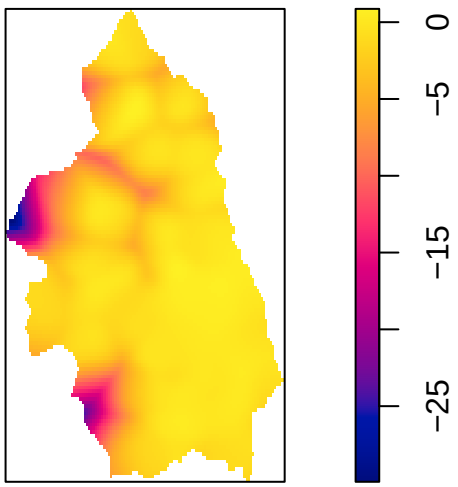
the sensibility at large risks. That why the plot has to be corrected so now the colors can indicate a difference. Now the plot is more clear and can point out the risk zones. As without the log transformation we have a zone in the midle-lower right of the window with some zones near it, with special mention to the one we saw in the previous plots which locates in the upper midle part of the window.

Once the instensities have been analzed we shall proced with the densities (with the log transformation too as recomended enhancing the risk differences). For this we use the same procedure as before but will try to focous on the places where the probabilty is low.

```
# Log Densities

pcb <- risk(cases,control)
```

```
## Estimating case and control densities...Done.
```

```
# The plots of cases and controls remain the same. Will thus not show them again in the report.
# # Cases
# plot(pcb$f, ribbon=F)
# # Controls
# plot(pcb$g, ribbon=F)

# Relative risk
plot(pcb$rr, main="Relative risk")
```




```
max(pcb$rr)
```

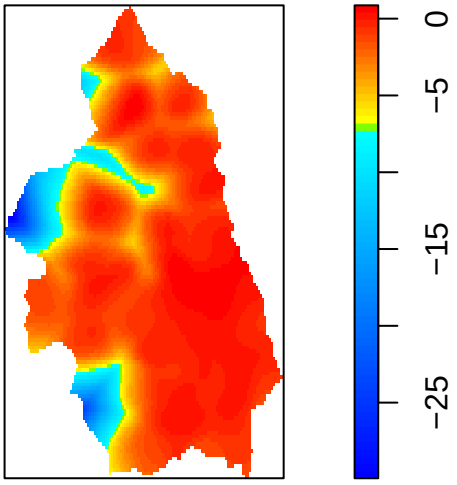
```
## [1] 0.8663849
```

```
min(pcb$rr)
```

```
## [1] -29.92377
```

```
plot(pcb$rr,col=beachcolours(c(-6,2), sealevel = 0),main="Relative risk corrected")
```

Relative risk corrected



As we can see only the positive diagonal corners and in the upper left side edge are areas of low risk.

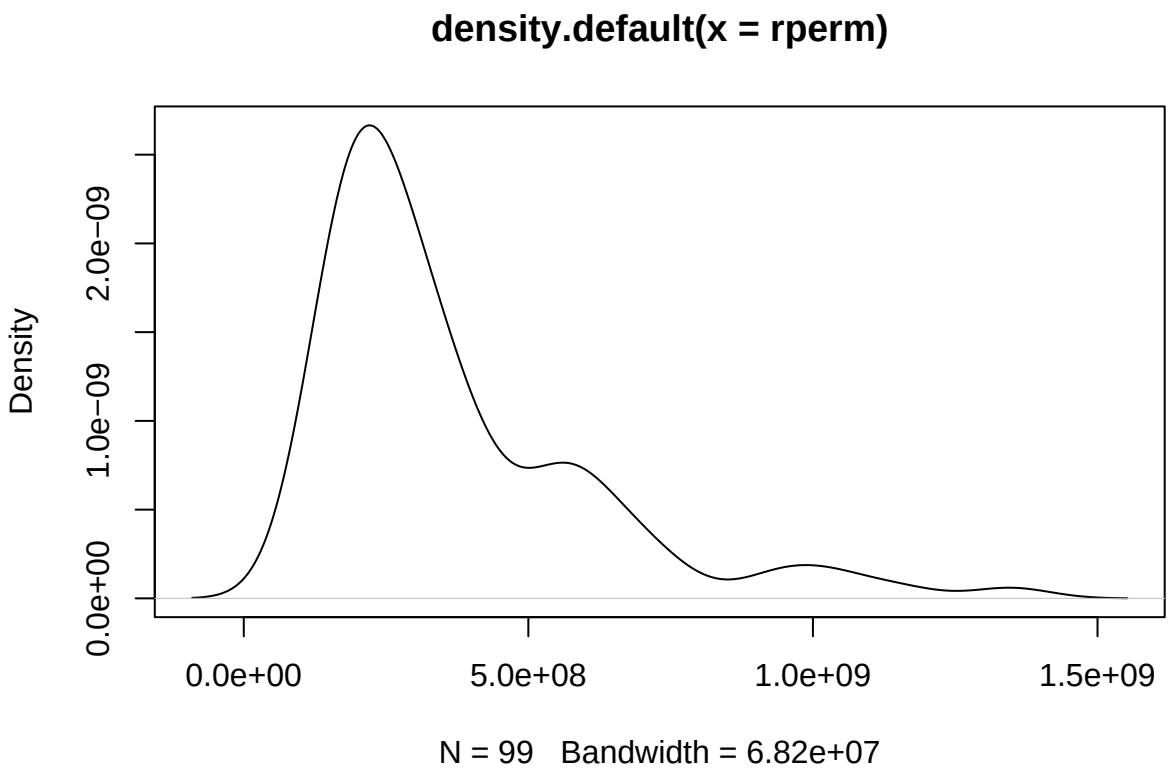
In order to know whether these variations are significant or not we need a base to compare. In this case the base is made via simulation, using a permutation test (Monte Carlo test). Hence we will be able to compare the observed kernel density ratios with the simulated ones.

```
rho0 <- 0
pcb <- risk(cases,control)

cellsize <-pcb$rr$xstep*pcb$rr$ystep
ratiorho <- cellsize*sum((pcb$rr$v-rho0)^2, na.rm=T)

#permutation func
perm_rr<-function(){
  new_pc <-rlabel(pbcdata_poly)
  new_cases <-split(new_pc)$case
  new_controls<- split(new_pc)$control
  new_pcb <- risk(new_cases, new_controls)
  cellsize <- new_pcb$rr$xstep*new_pcb$rr$ystep
  ratio_perm <- cellsize*sum((new_pcb$rr$v-rho0)^2, na.rm=T)
  ratio_perm
}
nsim<-99
set.seed(1234)
rperm <-sapply(1:99, function(i) perm_rr())
```

```
# P-value
(sum(rperm > ratiorho)+1)/(nsim+1)
```

```
## [1] 0.03
```

```
plot(density(rperm))
```

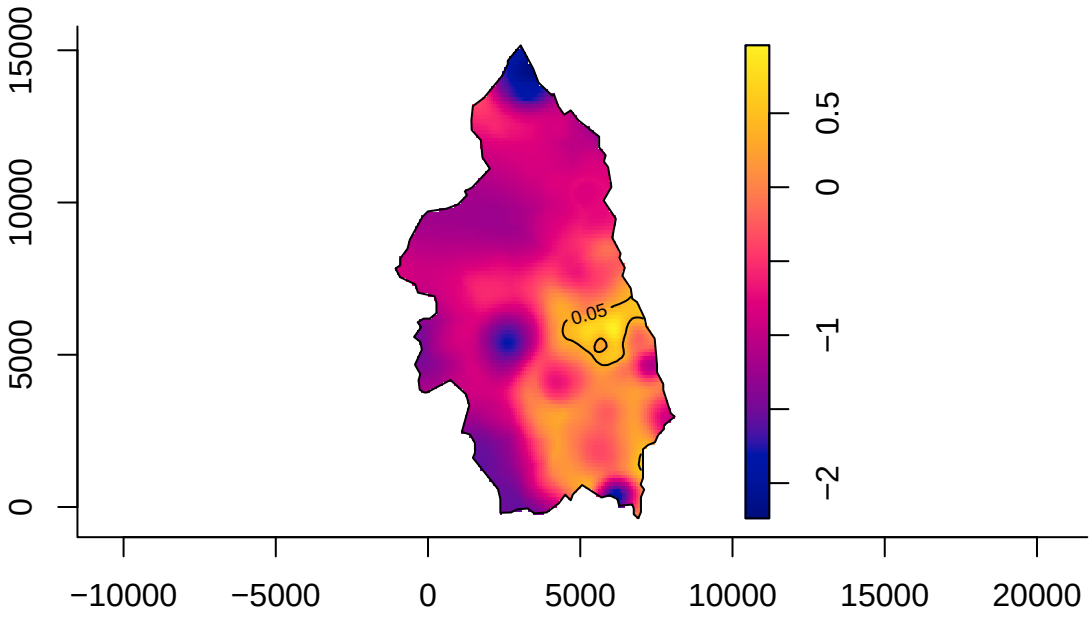



```
ratiorho
```

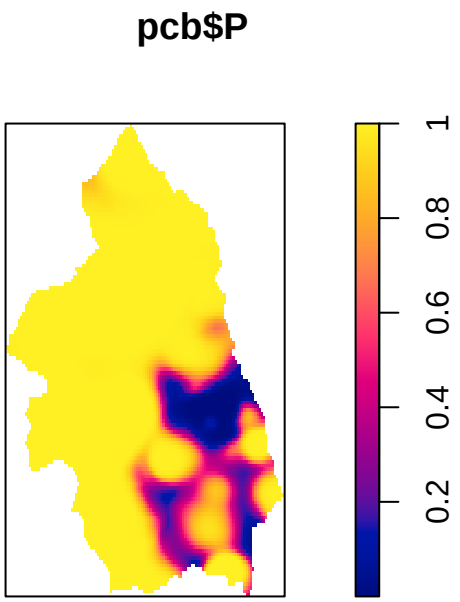
```
## [1] 1886921336
```

```
# P-values Contour Plot
pcb <- risk(cases,control,tolerate = T,adapt=T)
```

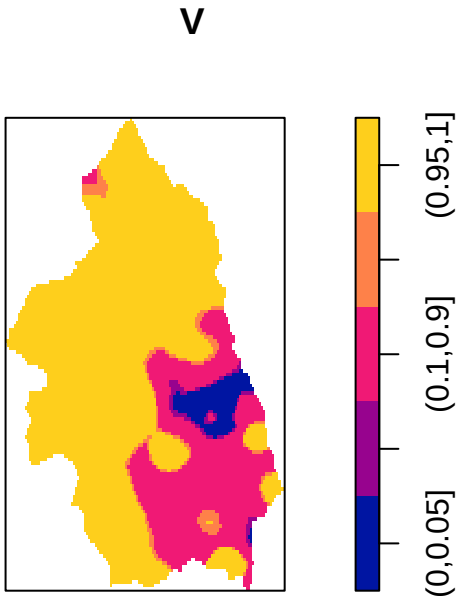
```
plot(pcb)
```



```
plot(pcb$P)
```



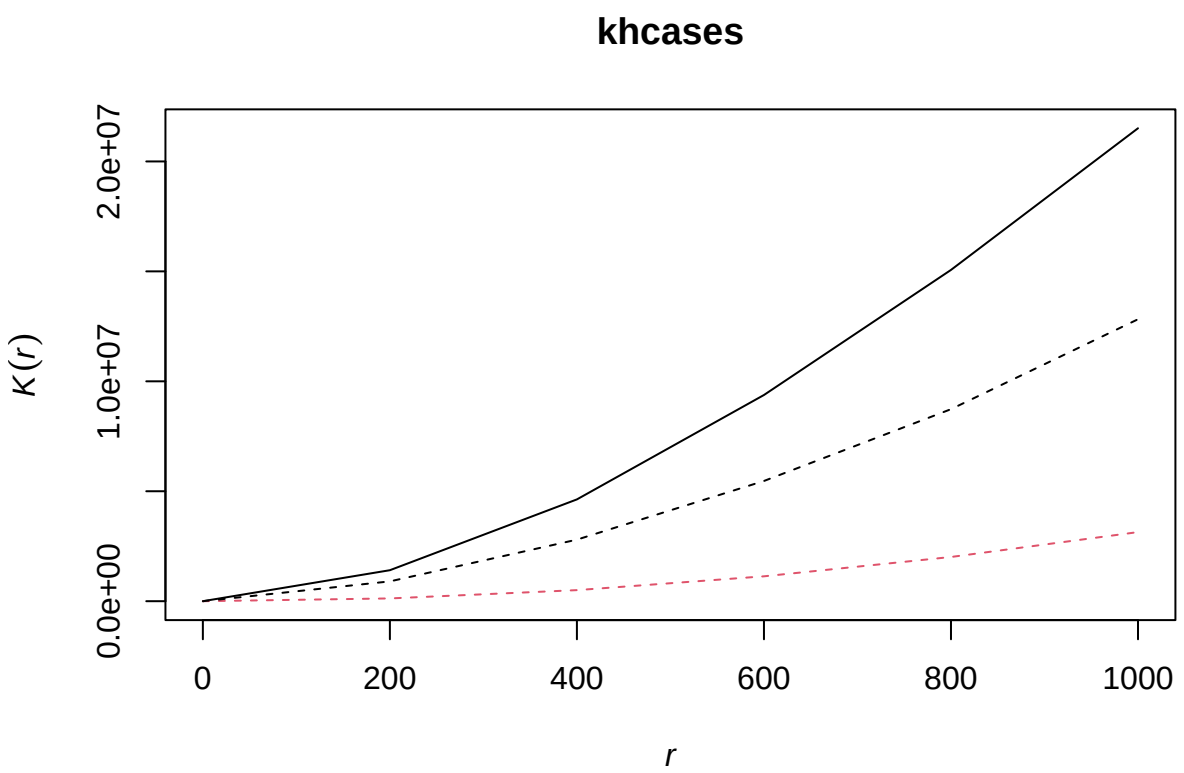

```
pcb_p<-cut(pcb$P,breaks=c(0,0.05,0.1,0.9,0.95,1))
V <- tess(image = pcb_p)
plot(V,valuesAreColours=FALSE)
```



One the permutation test is done we can see the mapping of the different p-values of the test done. To know if the model is significant or not. Here we see that the central part (the one with more point intensity) has considerably more cases since its p-value is so low. On the other hand he can not guarantee that the density is specially different in the rest of the map (or at least with the usual confidence level of 95%).

Now, in order to go on with our analysis, we should asses the general spatial clustering. In this case we are going to use a Poisson (random) process to compare if we have clustering or not. For this we are going to compare the expected number of points of the process within a certain distance using Ripley's K function. Besides we are going to use simulation in order to give an envelope.

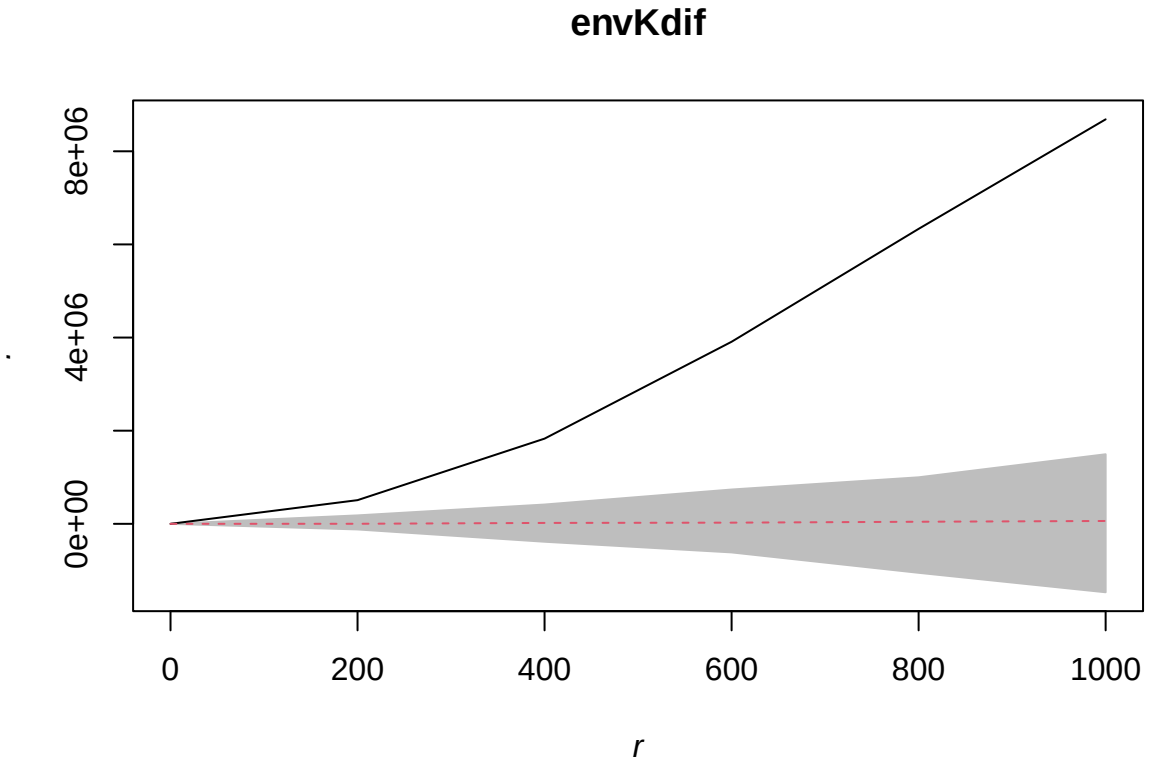
```r
s=seq(0,1000,by=200)
khcases<-Kest(cases, r=s, correction="iso")
khcontrols<-Kest(control, r=s, correction="iso")

plot(khcases, legend=F)
lines(khcontrols$r,khcontrols$iso,lty=2)
```




```r
# Difference of k functions with envelope
xppp=pbcdata_poly
Kdif <- function(xppp, r, cr="iso")
{
  k1 <- Kest(xppp[marks(xppp)=="case"], r=r, correction= cr)
  k2 <- Kest(xppp[marks(xppp)=="control"], r=r, correction= cr)
  D=k1[[cr]]-k2[[cr]]
  res <-data.frame(r=r, D=D)
  return(fv(res, valu="D", fname="D"))
}

nsim <- 39
envKdif <- envelope(pbcdata_poly, Kdif, r=s, nsim=nsim, nrank=1, savefuns=T, simulate=expression(rlabel
```

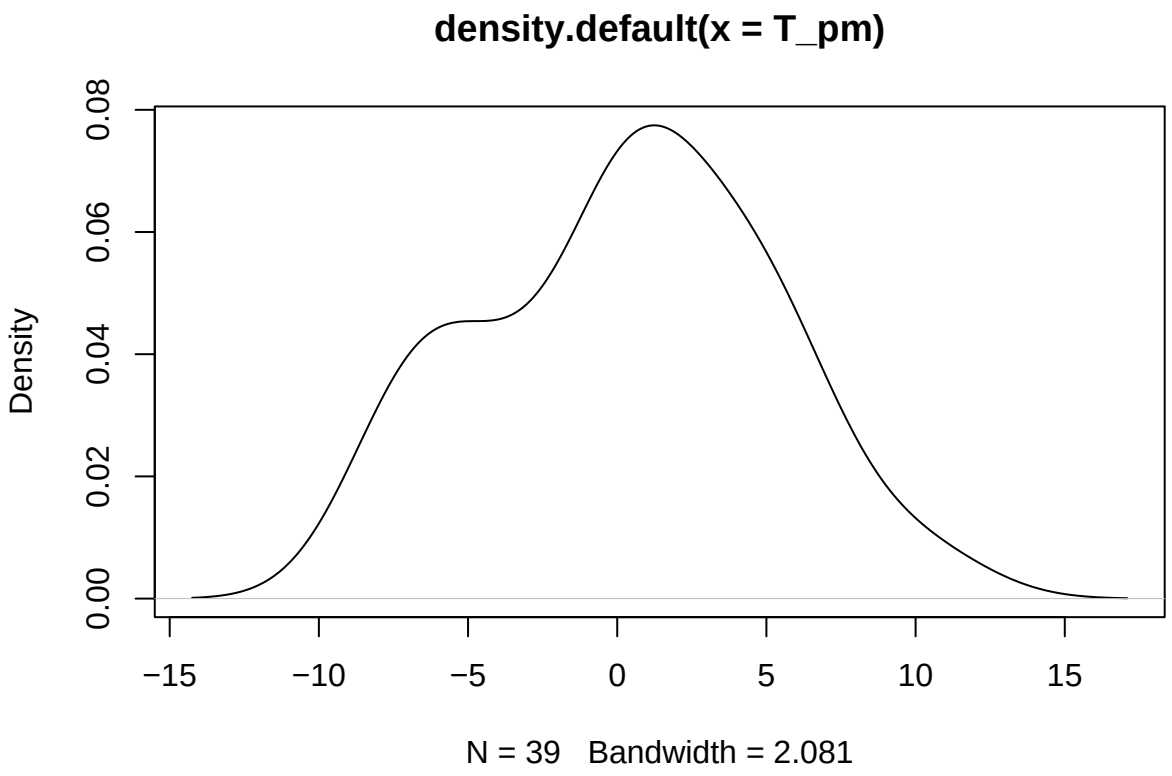
```
## Generating 39 simulations by evaluating expression  ...
## 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28
##
## Done.
```

```r
plot(envKdif, legend=F)
```



When the plot is obtained we can compare if our K is greater or not of the one computed using the Poisson process (random distributed data) as a reference. In this case we see that the continuous line (ours) is greater than the red one, therefore it signals that the is a clustering pattern (hence there is interaction). Now we can define a clear border to the envelope so we can look at the interval to make sure our hypothesis of clustering is enough evident. Finally in order to end the analysis we can have a look at the parameters of the simulations done at a certain bandwidth (very important) and again show the limits of the envelope of the K function done from the Poisson data. Finally we are going to test if the cases and controls have the same intensities hence having the same K functions. This is done by permutations.

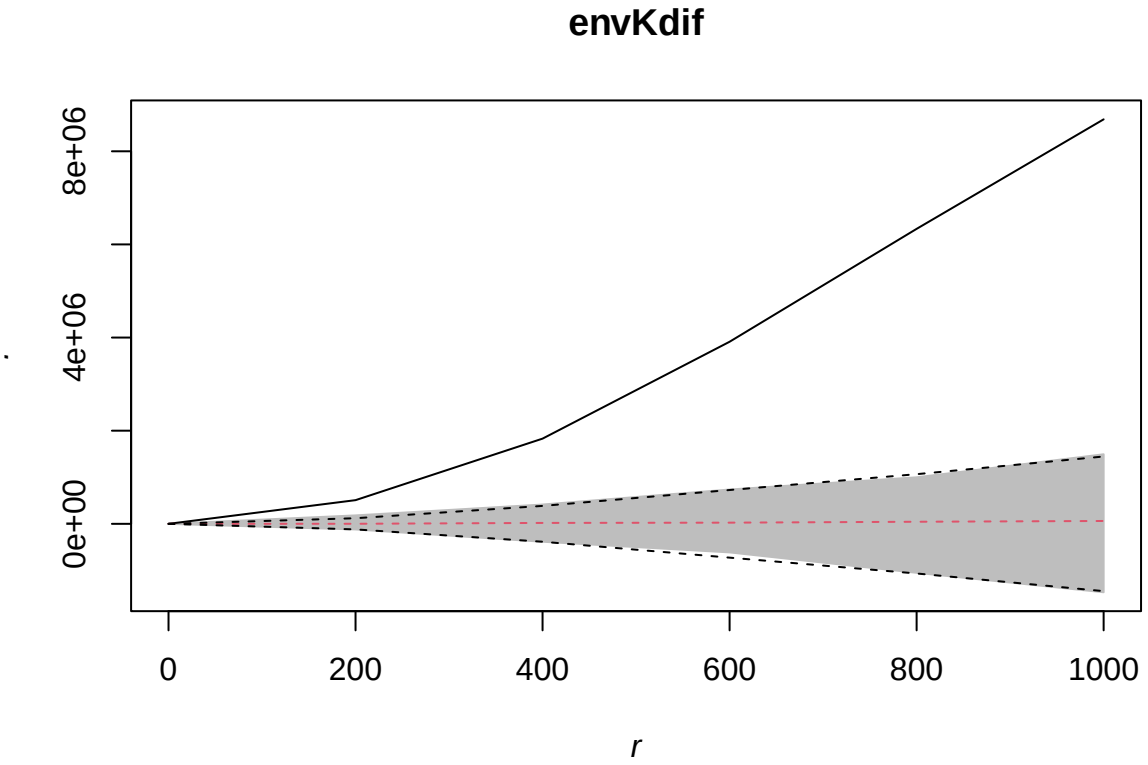
```
#Test

# Extract the simulated functions
simfuns<-as.data.frame(attr(envKdif, "simfuns"))[,-1]
#Compute diagonal of var-cov matrix of dif. K-functions
khcovdiag<-apply(simfuns, 1, var)

#Test
T0<-sum( ((khcases$iso-khcontrols$iso)/sqrt(khcovdiag))[-1])
T_pm<-apply(simfuns, 2, function(X){
  sum((X/sqrt(khcovdiag))[-1])
})

plot(density(T_pm))
abline(v=T0)
```




```
pvalue<-2*(sum(abs(T_pm)>abs(T0))+1)/(nsim+1)
pvalue
```

```
## [1] 0.05
```

```
# Alternative option for limits
plot(envKdif, legend=F)
lines(s, -1.96*sqrt(khcovdiag), lty=2)
lines(s, +1.96*sqrt(khcovdiag), lty=2)
```



As seen with the p-value thecases and control have different K functions and therefore different intesisites therefore reafirming the results obtained above (clustering, risk variation).

So all in all, we have look at the data from primary biliary cirrhosis cases and compared that to a data of control to see if there were areas with higher risk of disease using spatial point patterns analysis. Throughout the analysis we have find clustering and difference in risks concluding that there is a certain area with significantly more incidence (and hence risk) of suffering from primary biliary cirrhosis.