Point_process_June9

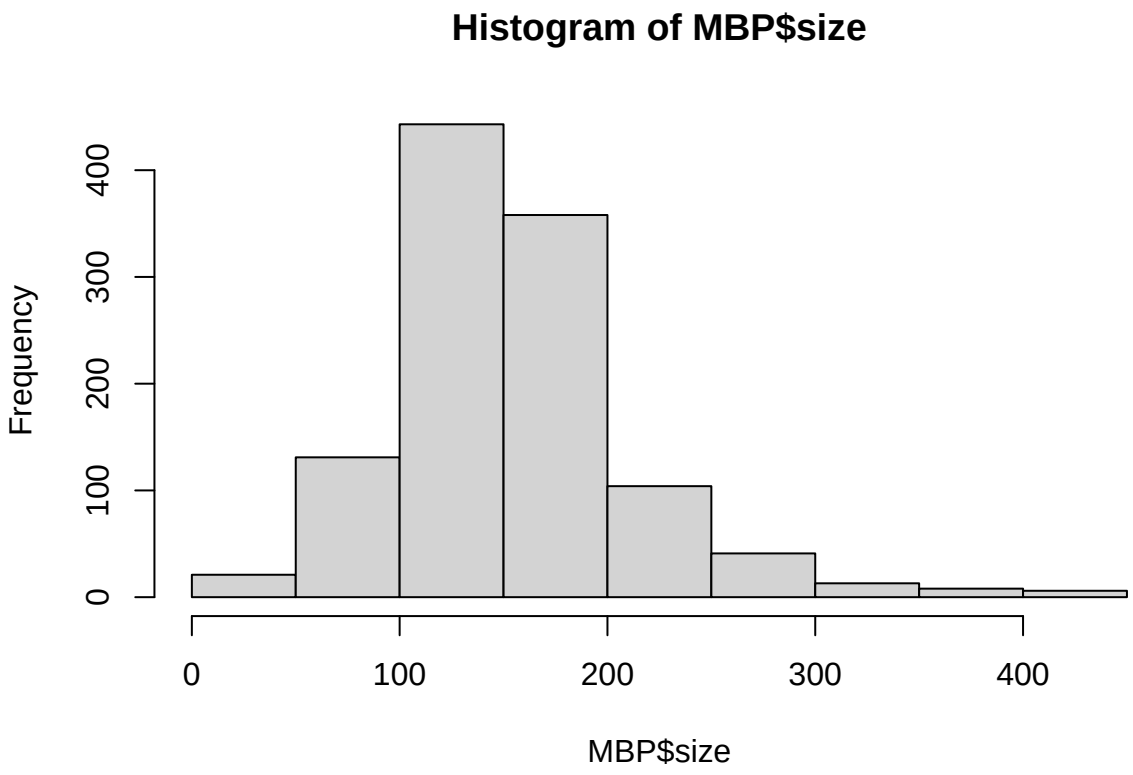
Hainan Xu

09/06/2022

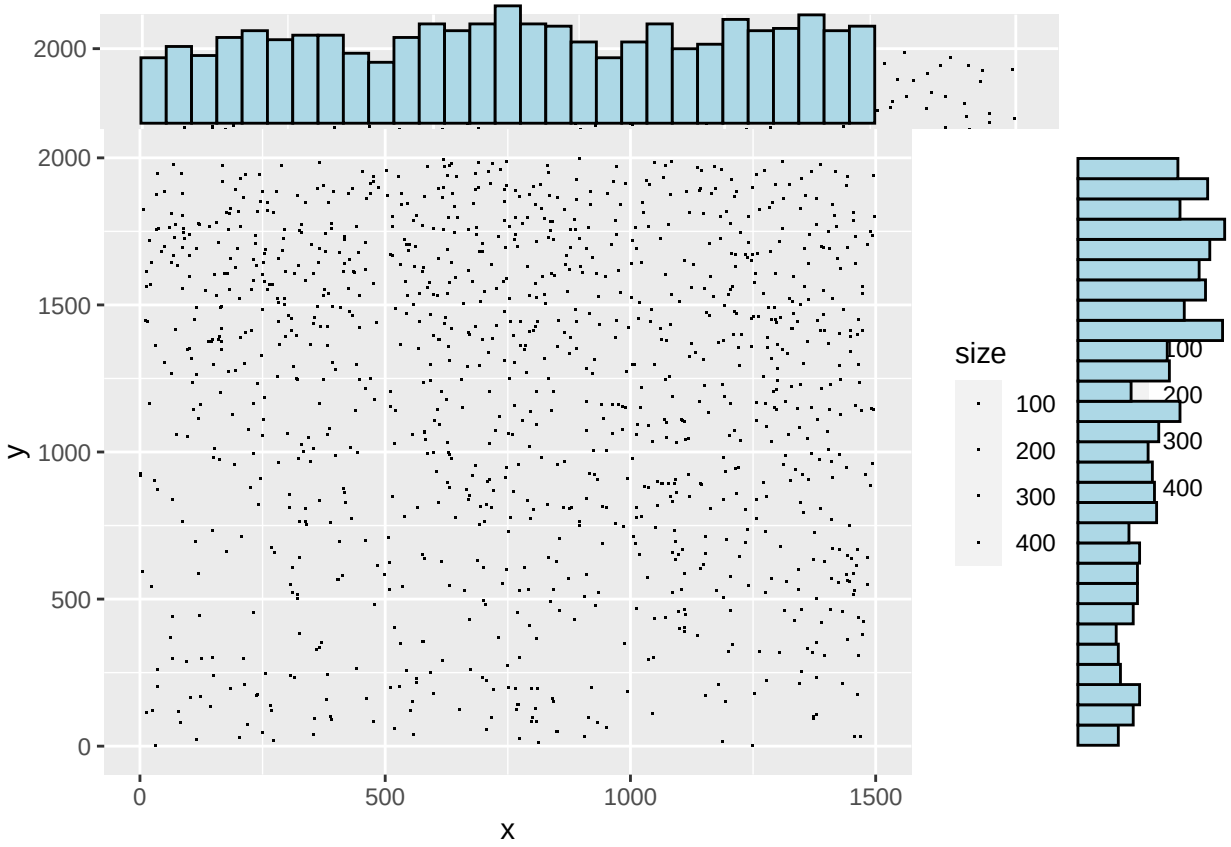
Point process

I converted the orginal dataset to the dataset with 4 variables: x position, y position, roi_scource,size of the rectangle.

Here is the size distribution of all the points. It shows a right-skewed distribution.



I plot the points based on thier position and the with marginal distributions of the points. More formatting need to be adjusted.


2000
2000
1500
1000
500
0
y
0
500
1000
1500
x
size
100
200
300
400
100
200
300
400

Shape Then, I converted the data as a marked point process. The mark I use is the size of the rectangle. After that, I extracted the shape of the visual cortex based on the points provided. Here `plot(ln)` can plot the whole marked point process, the size of the circle indicate the size of the original rectangle size.

```
## Marked planar point pattern: 1125 points
## marks are numeric, of storage type  'double'
## window: rectangle = [2, 1498] x [3, 1998] units
```

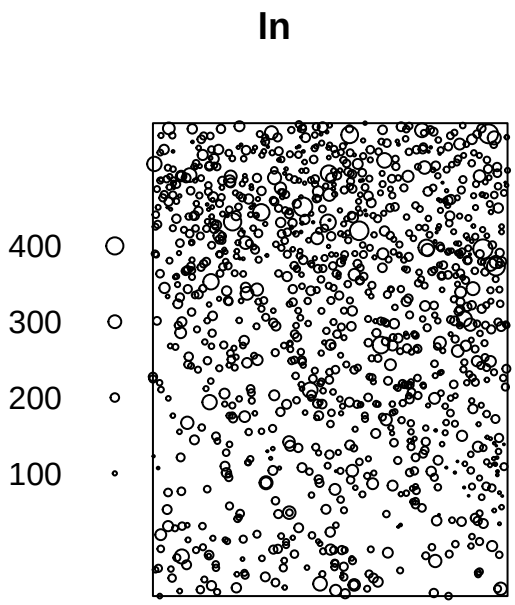


Figure 1: Marked Point Process

```
## Marked planar point pattern: 1125 points
## marks are of storage type  'character'
## window: polygonal boundary
## enclosing rectangle: [2, 1498] x [3, 1998] units
```

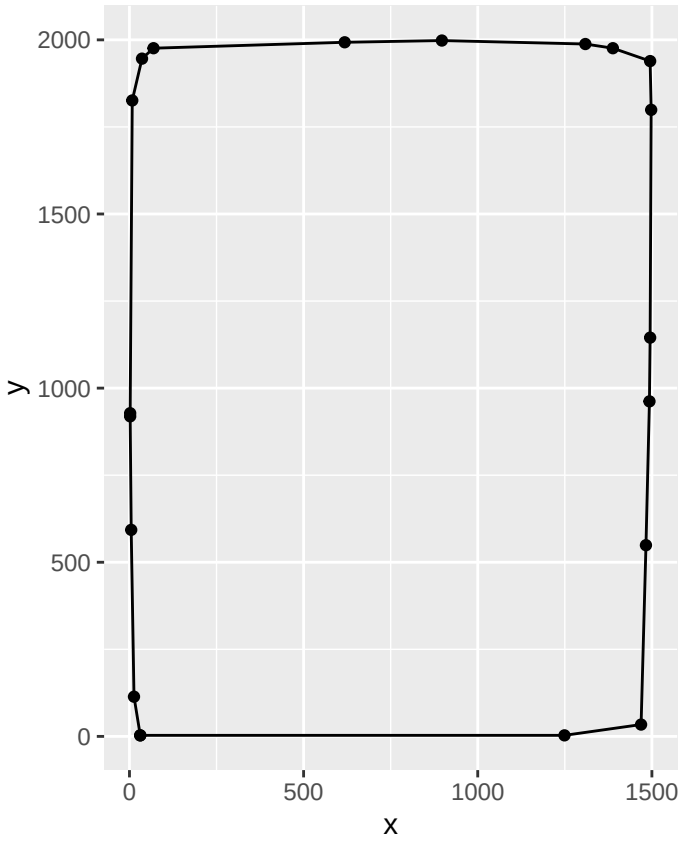


Figure 2: Marked Point Process

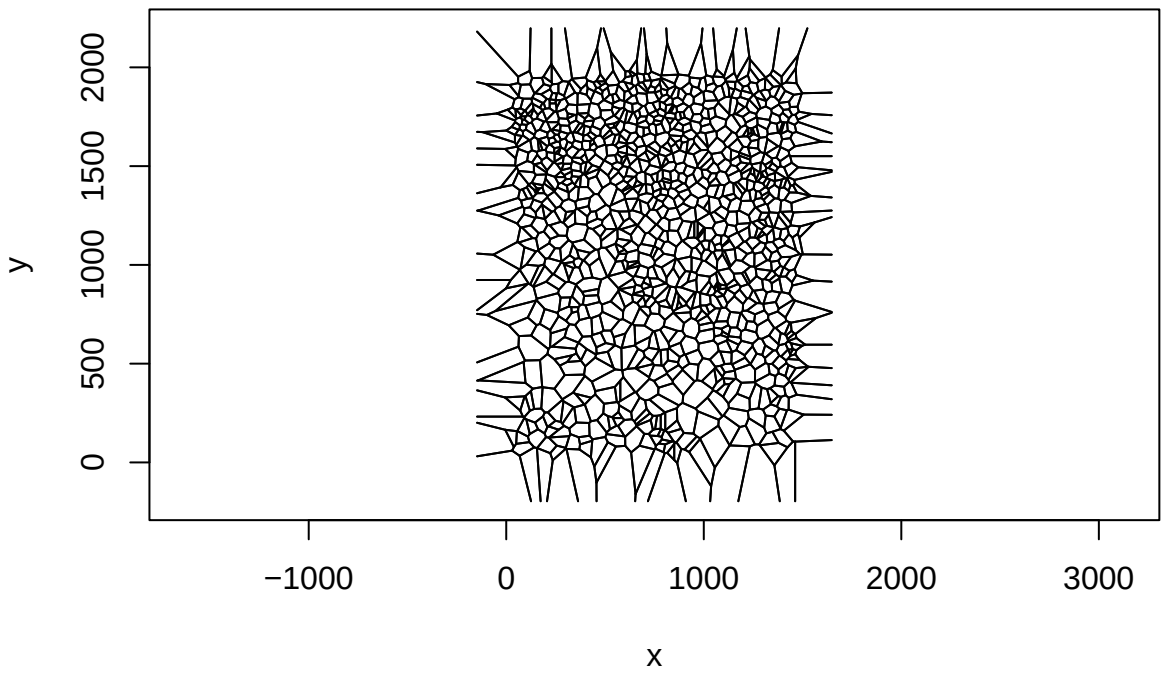
Voronoi Tessilation This is the Voronoi tessilation using `deldir` package.

```
library(deldir)
```

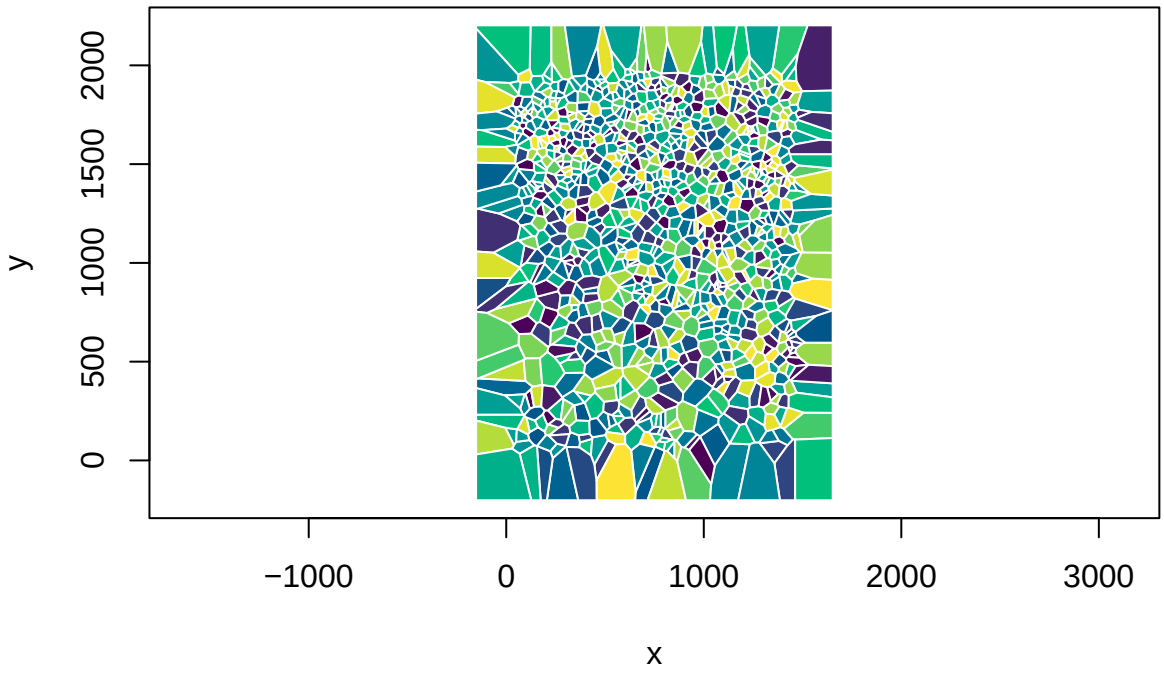
```
## deldir 1.0-6      Nickname: "Mendacious Cosmonaut"
```

```
##
##      The syntax of deldir() has had an important change.
##      The arguments have been re-ordered (the first three
##      are now "x, y, z") and some arguments have been
##      eliminated.  The handling of the z ("tags")
##      argument has been improved.
##
##      The "dummy points" facility has been removed.
##      This facility was a historical artefact, was really
##      of no use to anyone, and had hung around much too
##      long.  Since there are no longer any "dummy points",
##      the structure of the value returned by deldir() has
##      changed slightly.  The arguments of plot.deldir()
##      have been adjusted accordingly; e.g. the character
##      string "wpoints" ("which points") has been
##      replaced by the logical scalar "showpoints".
##      The user should consult the help files.
```

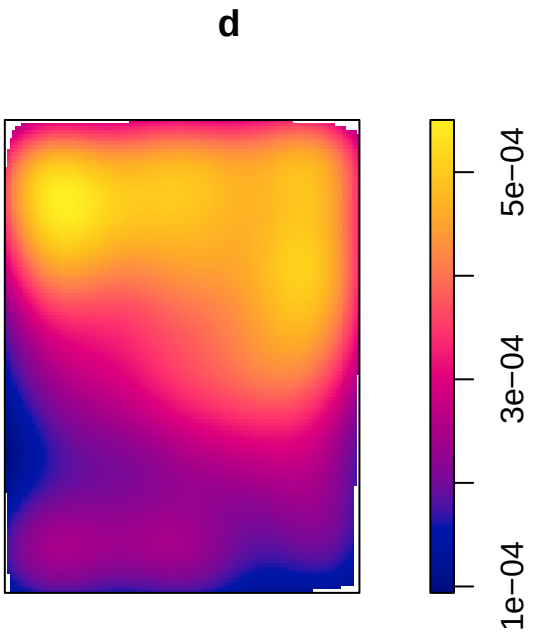
```
# Data

x <- MBP$x
y <- MBP$y

# Calculate Voronoi Tesselation and tiles
tesselation <- deldir(x, y)
tiles <- tile.list(tesselation)

plot(tiles,pch=10,showpoints = FALSE)
```




```
plot(tiles, pch = 1,
     col.pts = "white",
     border = "white",
     fillcol = hcl.colors(50, "viridis"),
    showpoints = FALSE
     )
```


y
0
500
1000
1500
2000
−1000
0
1000
2000
3000
x

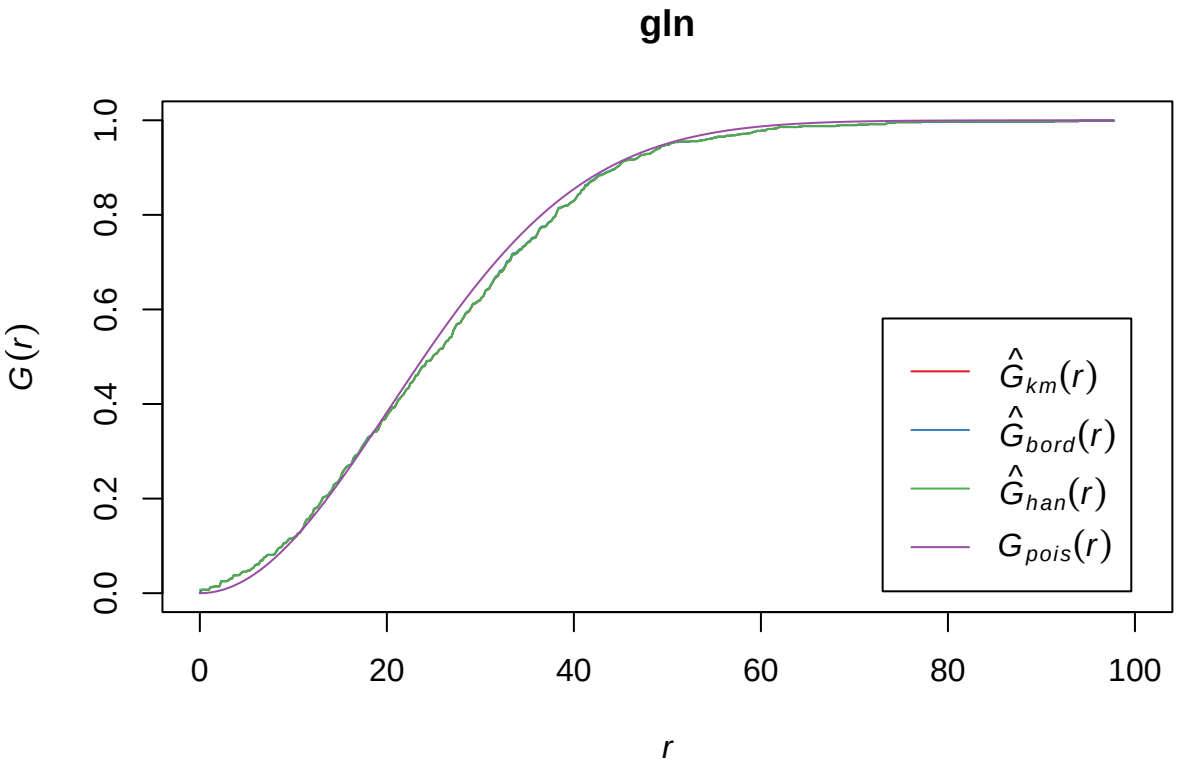
First order effect: estimation of the intensity. This is the estimation of the intensity of the MBP cells.



Second order effect: distance to it's nearest neighbor. Used 3 different edge effect correction. Here, only the correlation of margin is calculated since there are too much point.

G(r) is the cumulative distance, r is the distance.

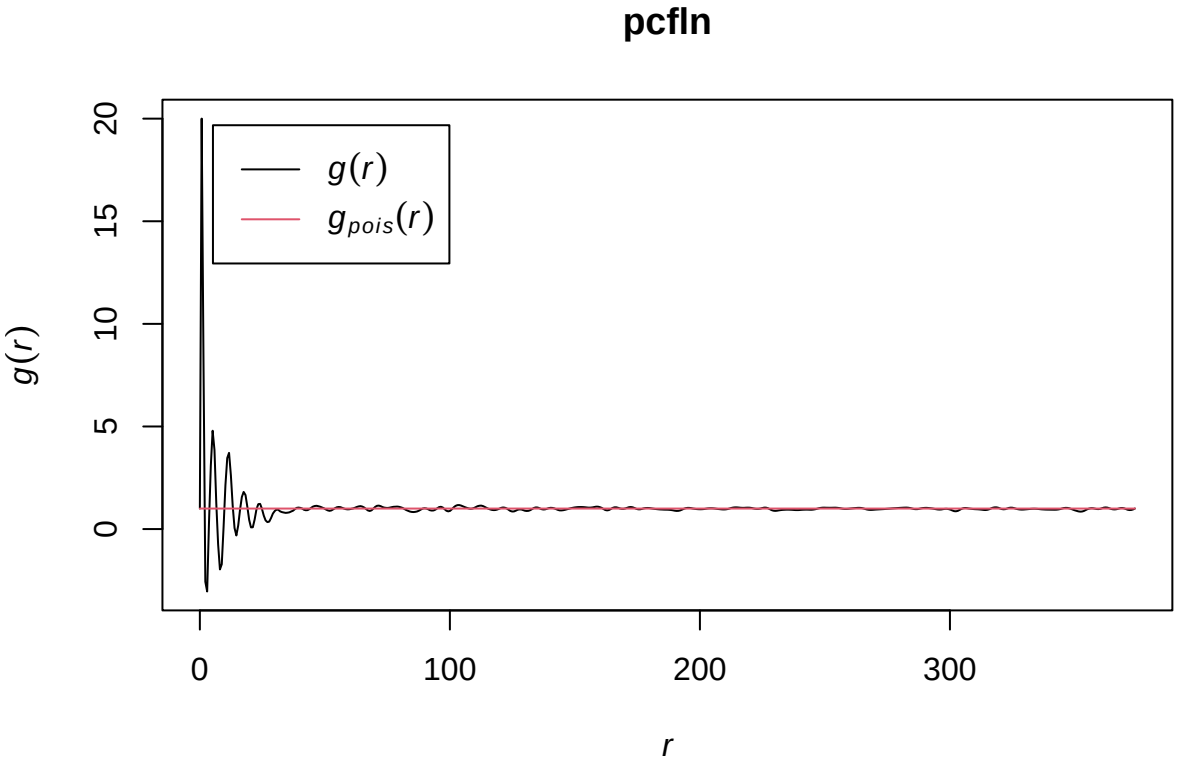
```
## Function value object (class 'fv')
## for the function r -> G(r)
## ......................................................................
##         Math.label      Description
## r       r               distance argument r
## theo    G[pois](r)      theoretical Poisson G(r)
## han     hat(G)[han](r)  Hanisch estimate of G(r)
## rs      hat(G)[bord](r) border corrected estimate of G(r)
## km      hat(G)[km](r)   Kaplan-Meier estimate of G(r)
## hazard  hat(h)[km](r)   Kaplan-Meier estimate of hazard function h(r)
## theohaz h[pois](r)      theoretical Poisson hazard function h(r)
## ......................................................................
## Default plot formula:  .~r
## where "." stands for 'km', 'rs', 'han', 'theo'
## Recommended range of argument r: [0, 44.48]
## Available range of argument r: [0, 97.741]
```

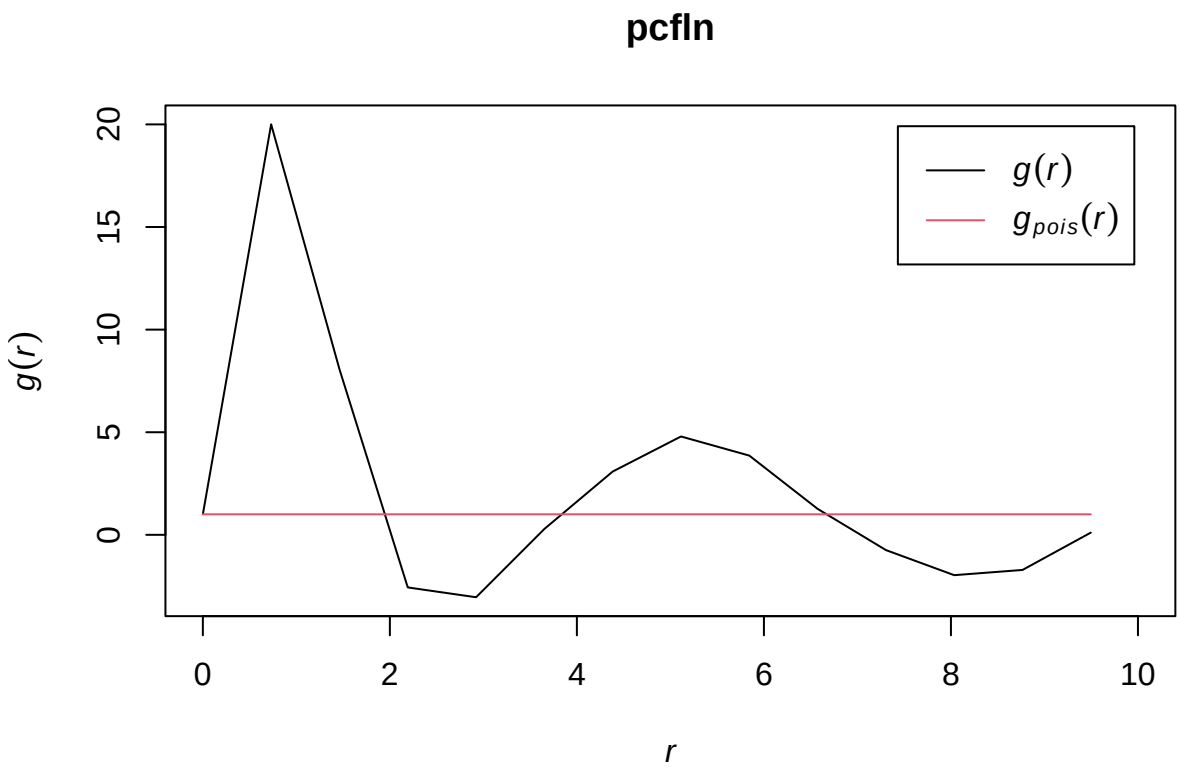


Paired correlation function

```
## Function value object (class 'fv')
```

```
## for the function r -> L[inhom](r)
## ......................................................................
##             Math.label
## r           r
## theo        {L[inhom]^{pois}}(r)
## border      {hat(L)[inhom]^{bord}}(r)
## bord.modif {hat(L)[inhom]^{bordm}}(r)
## trans       {hat(L)[inhom]^{trans}}(r)
## iso         {hat(L)[inhom]^{iso}}(r)
##             Description
## r           distance argument r
## theo        theoretical Poisson L[inhom](r)
## border      border-corrected estimate of L[inhom](r)
## bord.modif modified border-corrected estimate of L[inhom](r)
## trans       translation-correction estimate of L[inhom](r)
## iso         Ripley isotropic correction estimate of L[inhom](r)
## ......................................................................
## Default plot formula:  .~.x
## where "." stands for 'iso', 'trans', 'bord.modif', 'border', 'theo'
## Recommended range of argument r: [0, 374]
## Available range of argument r: [0, 374]
```




pcfln
g(r)
r
g(r)
g_pois(r)
0
5
10
15
20
0
2
4
6
8
10

Similar Analyais for a different pic:

```
MBP2<-read_csv(file.path("..", "data","H07-0500_79205589_179_2_MBP._analysis_results.csv"))%>%
  transmute(x=com_x,
            y=com_y,
            class="MBP")
```

```
## Rows: 1393 Columns: 16
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (5): roi_id, roi_source, roi_type, filename, analysis_date
## dbl (11): cell_number, com_x, com_y, pixel_area, background, mean_intensity,...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
MBP2
```

```
## # A tibble: 1,393 x 3
##        x     y class
##    <dbl> <dbl> <chr>
##  1  1345  1354 MBP
##  2   739   313 MBP
##  3  1245  1883 MBP
##  4   698   771 MBP
##  5   819  1306 MBP
##  6   911   701 MBP
##  7   439   642 MBP
##  8   730   746 MBP
##  9   552  1100 MBP
## 10   323    94 MBP
## # ... with 1,383 more rows
```

```
ln2 = with(MBP2,
  ppp(x = x, y = y, marks = class, xrange = range(x), yrange = range(y)))
```

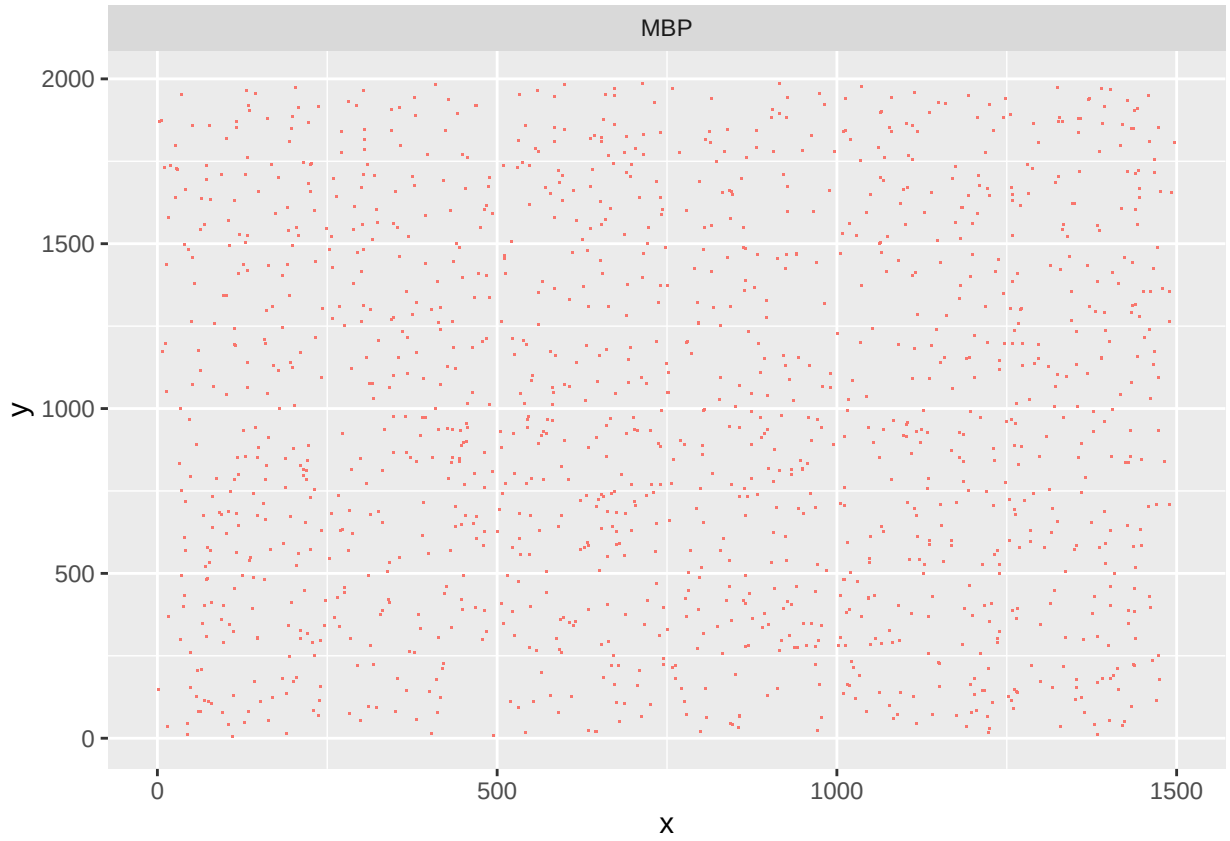
```
## Warning: data contain duplicated points
```

```
ln2
```

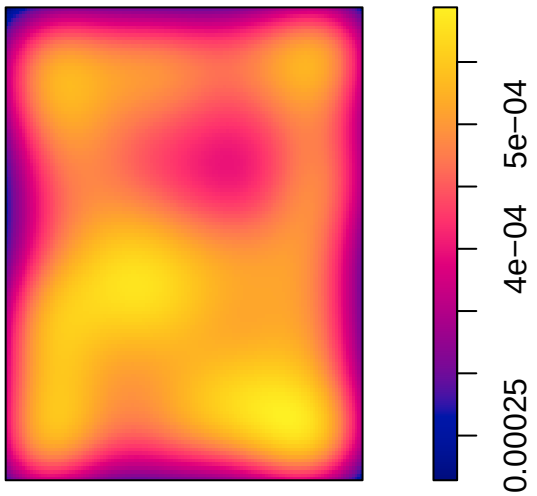
```
## Marked planar point pattern: 1393 points
## marks are of storage type  'character'
## window: rectangle = [2, 1497] x [6, 1987] units
```

```
ggplot(MBP2,
   aes(x = x, y = y, col = class)) + geom_point(shape = ".") +
   facet_grid( . ~ class) + guides(col =FALSE)
```

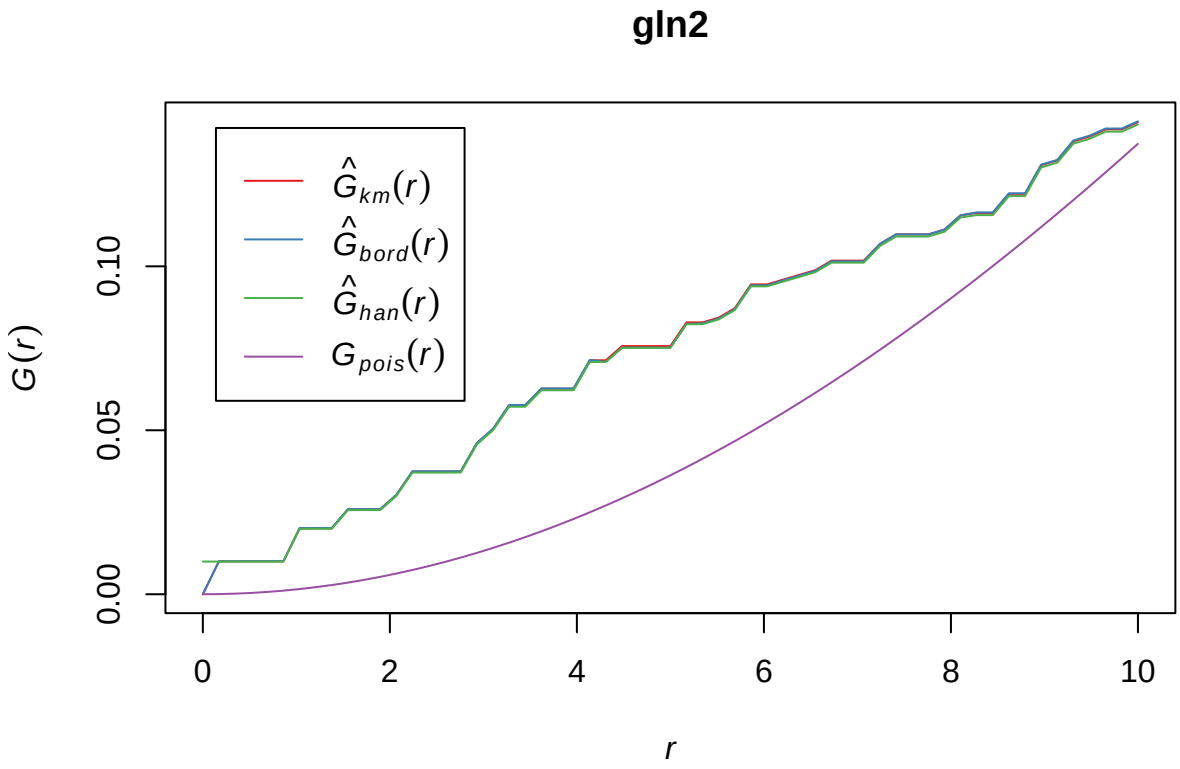
```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```




```
d2 = density(subset(ln2, marks == "MBP"), edge=TRUE, diggle=TRUE)
plot(d2)
```



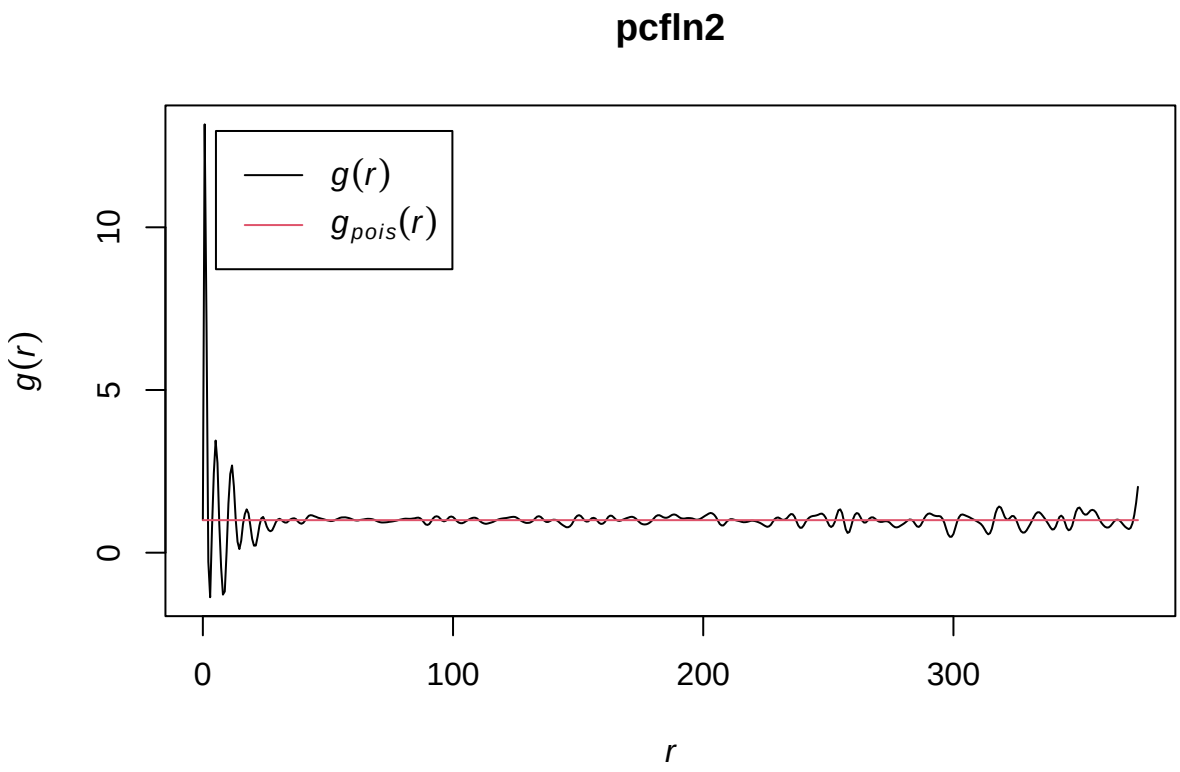

```
gln2 = Gest(ln2)


library("RColorBrewer")
plot(gln2, xlim = c(0, 10), lty = 1, col = brewer.pal(4, "Set1"))
```



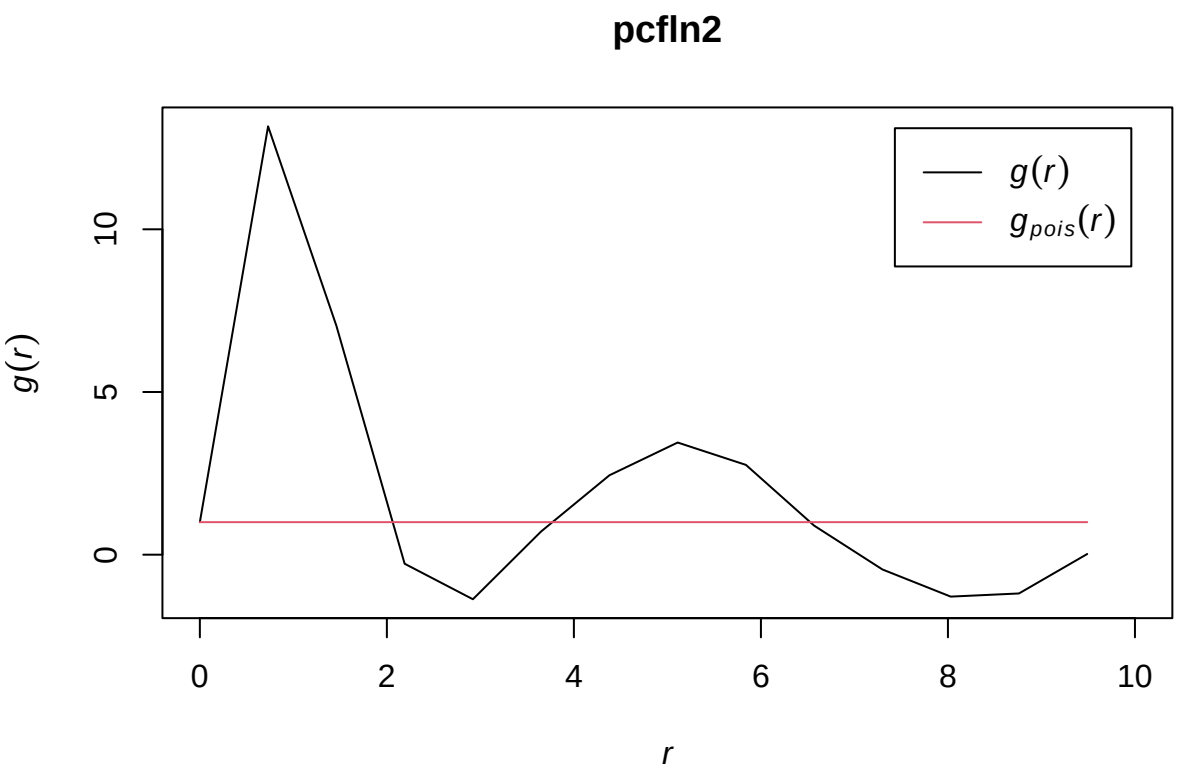

```
pcfln2 = pcf(Kinhom(subset(ln2, marks == "MBP")))
```

```
## number of data points exceeds 1000 - computing border correction estimate only
```

```
plot(pcfln2, lty = 1)
```




```
plot(pcfln2, lty = 1, xlim = c(0, 10))
```


pcfln2
g(r)
g_pois(r)
0
5
10
g(r)
0
2
4
6
8
10
r