

# Assignment 1.2 : Computer Lab

Sungil Kim

## *Contents*

1. Data Description
2. Complete Spatial Randomness, Regularity, and Clustering
3. First-Order Effects: Estimation
  - 3.1 Quadrat counts
  - 3.2 Kernel estimation
4. First-Order Effects: Inference
  - 4.1 Quadrat count test for CSR
  - 4.2 Indices for Quadrat Count Data
5. Second-Order Effects: Estimation
  - 5.1 G and F functions
  - 5.2 The K function
6. Appendix

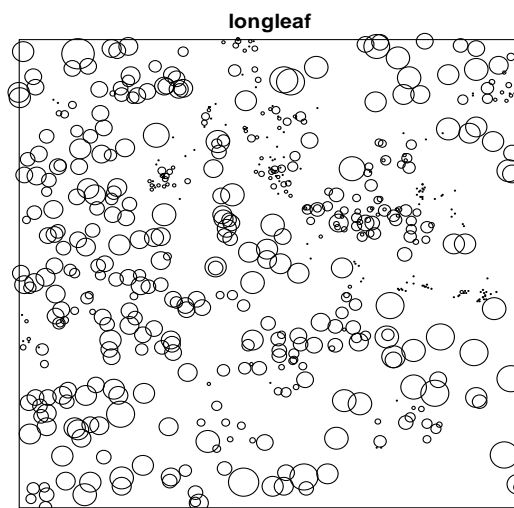
## 1. Data Description

Data consist of the coordinates and diameters (at breast height) of all longleaf pine trees at least 2 cm in diameter at breast height in 4 ha (200m\*200m) of forest in 1979.

```
>library(spatstat)
>data(longleaf)

marked planar point pattern: 584 points
marks are numeric, of type 'double'
window: rectangle = [0, 200] x [0, 200] metres

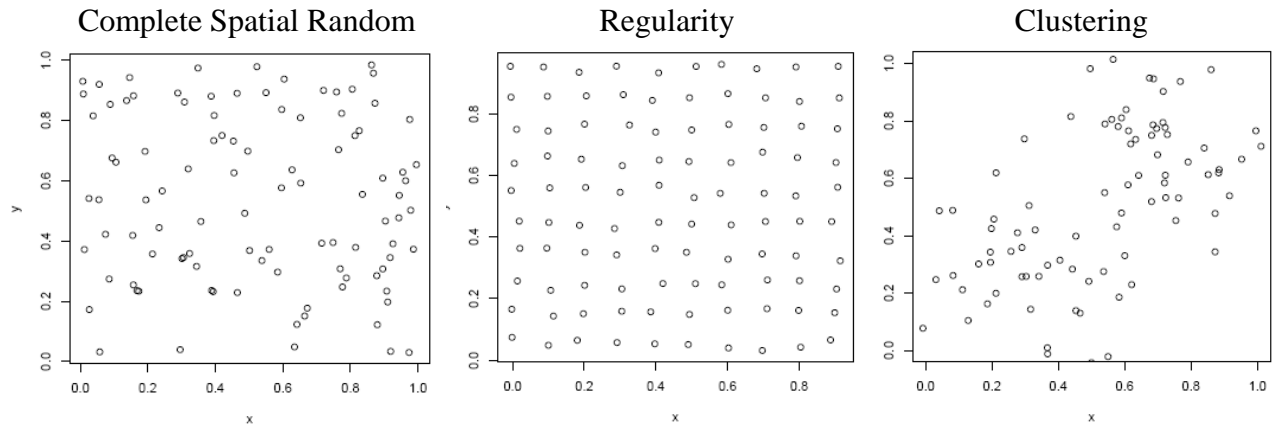
>plot(longleaf)
      0      20      40      60      80
0.000000 1.722522 3.445045 5.167567 6.890090
```



**Question: Do the spatial locations of the above figure appear completely spatially random? Or, are they regular, clustered?**

## 2. Complete Spatial Randomness, Regularity, and Clustering

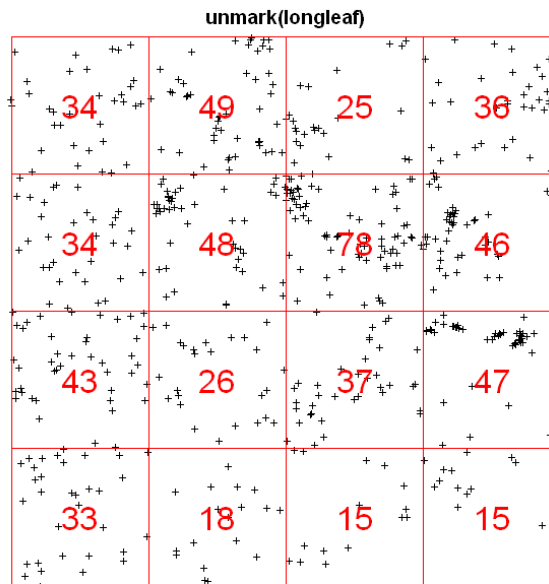
Complete spatial randomness = homogeneous Poisson Process



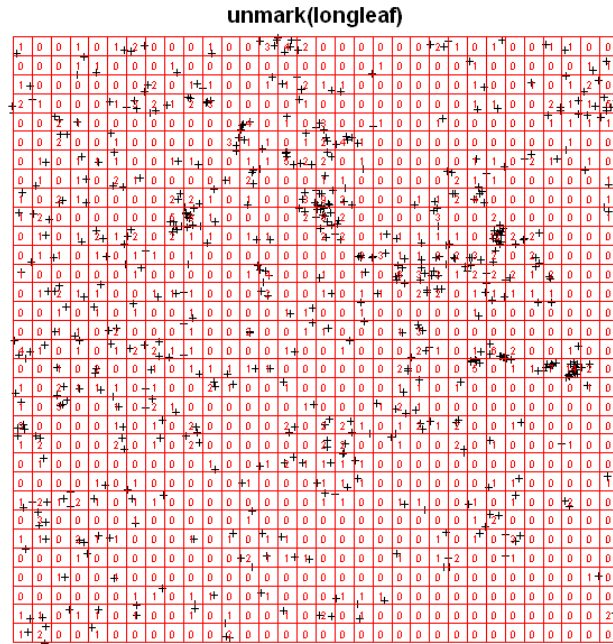
## 3. First-Order Effects: Estimation

### 3.1 Quadrat counts

```
> n<-4  
> qX <- quadratcount(longleaf, n, n)  
> plot(unmark(longleaf), pch="+")  
> plot(qX, add=TRUE, col="red", cex=2, lty=1)
```



What about n=32?



$$\lambda_h(s) = P(N(s, h) > 0) / h^2$$

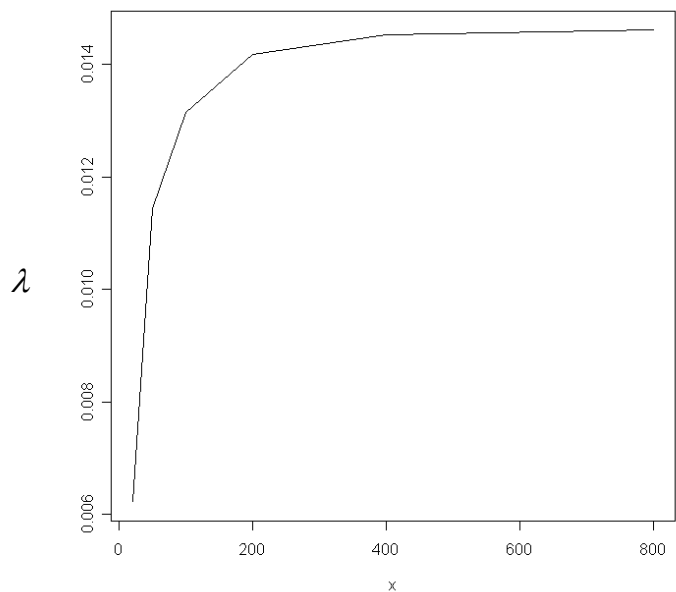
```
> (sum(qX>0)/n^2)/((200/n)^2)
[1] 0.009225
```

$$\lambda_h(s) \rightarrow \lambda(s) \text{ as } h \rightarrow 0$$

What if n=200?

```
> (sum(qX>0)/n^2)/((200/n)^2)
[1] 0.014175
```

```
lambda<-function(n){
qX <- quadratcount(longleaf, n, n)
(sum(qX>0)/n^2)/((200/n)^2)}
x<-c(20,50,100,200,400,800)
for(i in 1:6)
  y[i]<-lambda(x[i])
plot(x,y,type="l")
```



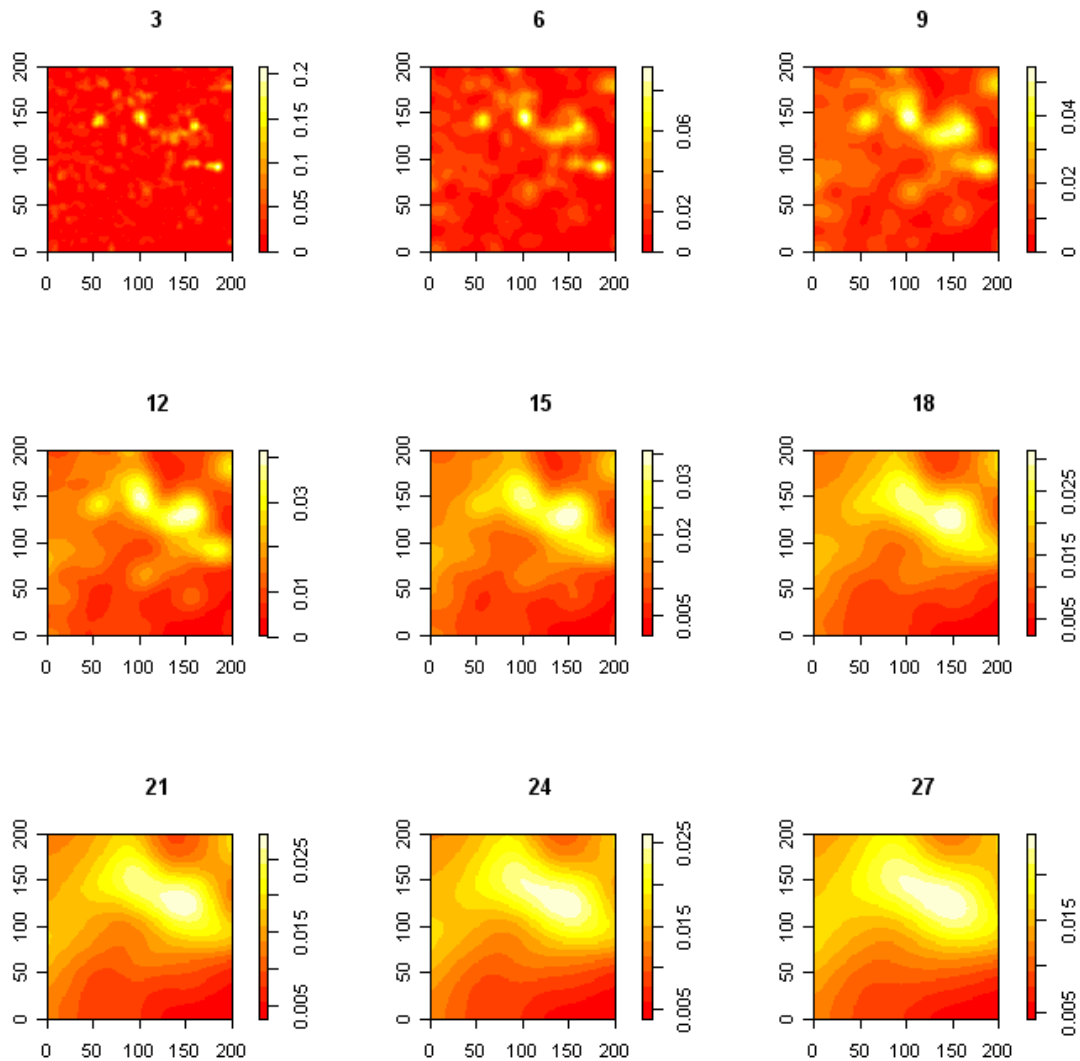
### 3.2 Kernel estimation

Suppose that the region of interest  $A$  is divided into regularly spaced square quadrats of size  $h \times h$ . For large  $h$ , the resulting quadrat counts destroy spatial information. However, they do give a global idea of subregions with high or low numbers of events per unit area. For small  $h$ , more spatial information is retained, but at the expense of a “busier” picture of numbers of events per unit area.

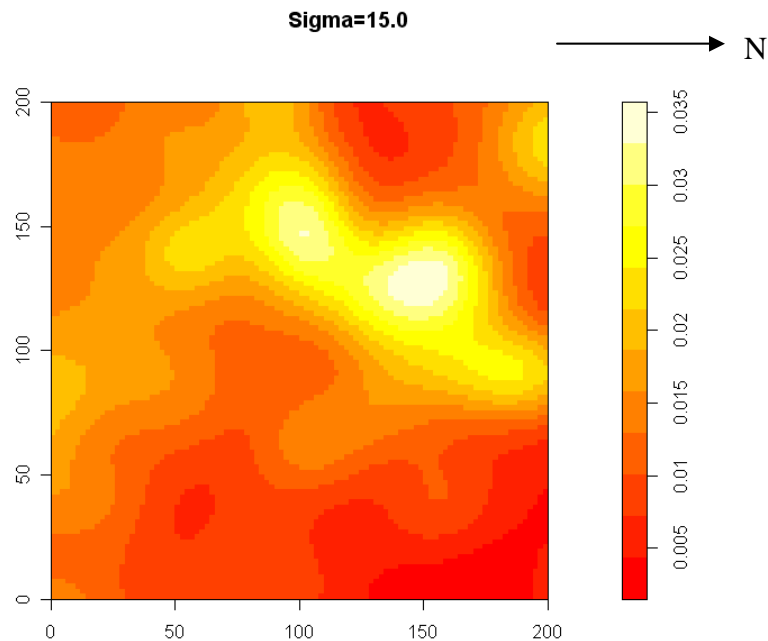
How is the kernel estimates of the intensity used with a single dataset?

Compare observed behavior versus what we expected to see under the assumption of complete spatial randomness (CSR). Intensity functions should be roughly constant.

```
par(mfrow=c(3,3))  
for(i in 1:9) plot(density.ppp(longleaf, 3*i), main=3*i)
```



```
par(mfrow=c(1,1))
plot(density.ppp(longleaf, 15),main="Sigma=15.0")
```



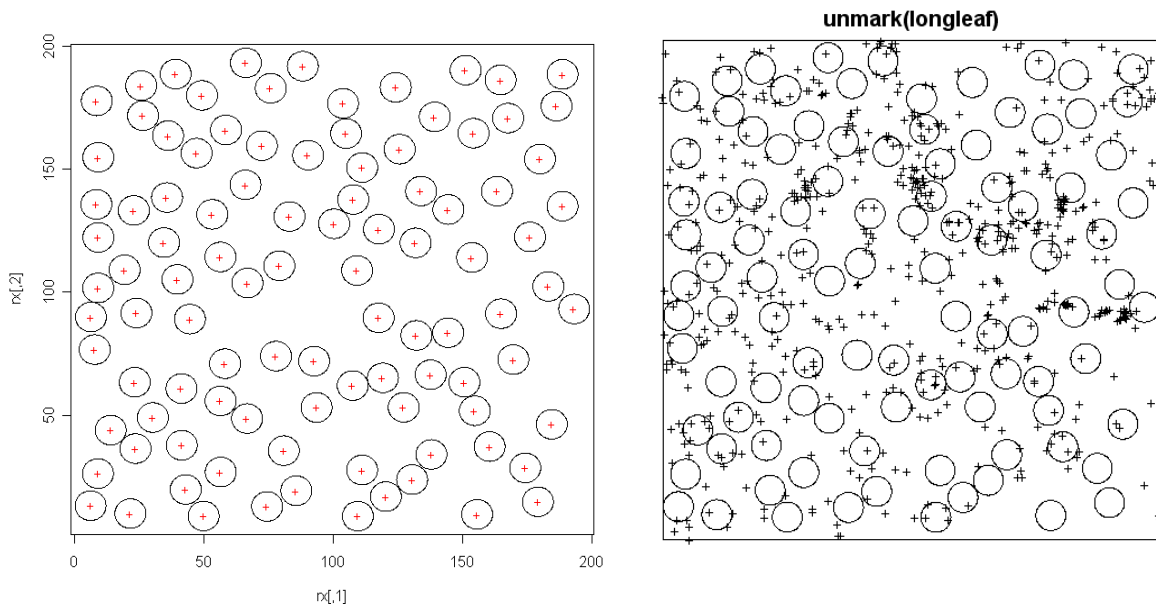
This figure suggests that the point pattern is non-stationary; there is a clear trend of increasing intensity from the eastern to the western half of the eastern to the western half of the study region. Peak intensities of more than  $0.025 \text{ m}^{-2}$  are found in a band extending from the west-central to the north-central part of the study region and in the northwestern corner. Intensities in the eastern portion are generally less than  $0.015 \text{ m}^{-2}$ .

## 4. First-Order Effects: Inference

### 4.1 Quadrat count test for CSR

We consider 100 circular quadrats, 6 m in radius. The Quadrat locations were random, but constrained so that no two quadrats overlap.

```
##100 random circular quadrats(not overlaped)##
rx<-matrix(100,100,2)
rx[1,1]<-runif(1,6,194)
rx[1,2]<-runif(1,6,194)
rx[2,1]<-runif(1,6,194)
rx[2,2]<-runif(1,6,194)
for (i in 1:100){
  while(sum(nndist(rx[1:i,])<12)>0){
    rx[i,1]<-runif(1,6,194)
    rx[i,2]<-runif(1,6,194)
  }
}
plot(rx,pch="+",col=2)
for(j in 1:100){
plot(disc(6,rx[j,]),add=TRUE)}
```



```

> count
[1] 7 1 2 0 2 0 1 0 0 4 3 4 0 8 1 6 2 0 1 5 0 2 0 0 2
[26] 0 10 1 1 0 2 0 0 1 3 1 2 0 0 2 1 5 0 0 0 0 3 2 5 1
[51] 3 3 0 0 0 1 1 5 3 5 0 3 0 1 7 2 2 1 0 2 0 4 1 3 2
[76] 1 2 1 4 1 3 2 1 0 1 1 0 2 1 2 1 1 2 0 1 0 1 1 0 1

>
> O
[1] 31 28 18 9 4 5 1 2 1 0 1

> E
[1] 2.393089e+01 3.422118e+01 2.446814e+01 1.166315e+01 4.169575e+00
[6] 1.192498e+00 2.842121e-01 5.806048e-02 1.037831e-02 1.648998e-03
[11] 2.358067e-04

```

$$f(k, \lambda) = \frac{\lambda^k e^{-\lambda}}{k!},$$

Trees per Quadrat	Observed Frequency	Expected Frequency ( $\lambda=1.43$ )
0	31	23.93
1	28	34.22
2	18	24.47
3	9	11.66
4	4	4.17
5	5	1.19
6	1	0.28
7	2	0.06
8	1	0.01
9	0	0.00
10	1	0.00

$$\chi_{n-1}^2 = \sum_{i=1}^n \frac{(O_i - E_i)^2}{E_i}$$

```

> T<-matrix(0,6,2)
> T[1:5,1]<-O[1:5]
> T[6,1]<-sum(O[6:11])
> T[1:5,2]<-E[1:5]
> T[6,2]<-sum(E[6:11])
> pearson<-sum((T[,1]-T[,2])^2/T[,2])
> pvalue<-1-pchisq(pearson,5)
> pvalue
[1] 6.125843e-10

```

➔ Pearson's chi-square test indicates significant departure from a Poisson distribution. Thus, complete spatial randomness (csr) is rejected

## 4.2 Indices for Quadrat Count Data

Once complete spatial randomness (csr) is rejected, the next step in a spatial analysis may be to measure the departure from csr. Various indices that have appeared in the literature were calculated from the sample of 100 quadrats. For example,

$$I : \frac{S^2}{\bar{X}}$$

$$ICS : \frac{S^2}{\bar{X}} - 1$$

$$ID : \frac{\sum_{i=1}^n (X(A_i) - \bar{X})^2}{n\bar{X}}$$

```

count_mean<-mean(count)
count_var<-var(count)
I<-count_var/count_mean
ICS<-count_var/count_mean-1
ID<-sum((count-count_mean)^2)/100/count_mean

> I
[1] 2.251931
> ICS
[1] 1.251931
> ID
[1] 2.229412

```

Values of I greater than 1 and ICS greater than 0 indicate that the longleaf pines are clustered. If ICS were less than 0, then this would indicate a tendency for regular spacings. ID>1 indicates clustering, and ID<1 regularity.

## 5. Second-Order Effects: Estimation

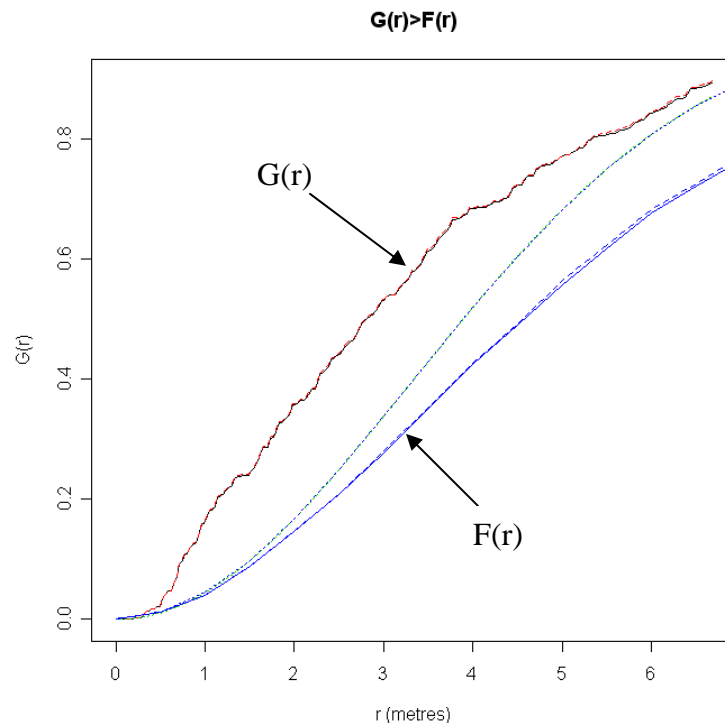
### 5.1 Distance Method ( $G$ and $F$ functions)

The distance methods make use of precise information on the locations of events and have the advantage of not depending on arbitrary choices of quadrat size or shape.

#### $G$ vs $F$

Define  $G(r)$  to be the probability that the distance from a randomly chosen event to its nearest event is less than or equal to  $r$ . Likewise, define  $F(r)$  to be the probability that the distance from a randomly chosen point to its nearest event is less than or equal to  $r$ .

```
plot(Gest(longleaf),main="G(r)>F(r)")
plot(Fest(longleaf),add=TRUE,col="blue")
```



Under CSR,  $G$  and  $F$  are similar, both follows (green line)

$$G(r) = 1 - \exp(-\lambda\pi r^2), \quad r \geq 0,$$

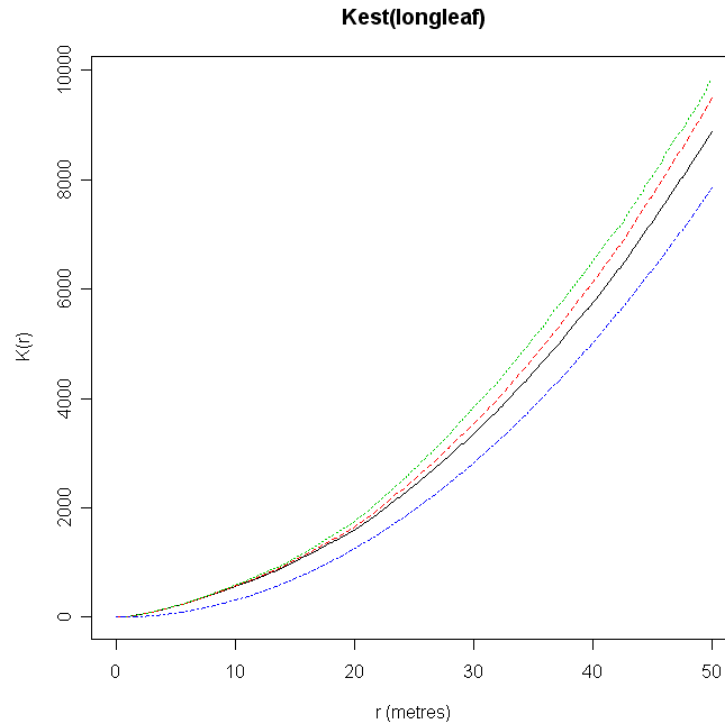
Under clustered pattern, we have that  $G(r) > F(r)$ .

Under regular pattern, we have that  $G(r) < F(r)$ .

## 5.2 The K function

The problem with G and F is that they use distances only to closest events, i.e., consider only smallest spatial scales of the pattern. Furthermore, this scale depends on intensity. A more effective summary is an estimate of the K function.

```
> plot(Kest(longleaf))
      lty col
iso    1  1(black)
trans  2  2(red)
border 3  3(green)
theo   4  4(blue)
```



The estimation of K is hampered by edge effects arising from the unobservability of points of the random pattern outside the window. An edge correction is needed to reduce bias (Baddeley, 1998; Ripley, 1988). The corrections implemented here are

- border the border method or “reduced sample” estimator (see Ripley, 1988). This is the least efficient (statistically) and the fastest to compute. It can be computed for a window of arbitrary shape.
- isotropic/Ripley Ripley’s isotropic correction (see Ripley, 1988; Ohser, 1983). This is implemented for rectangular and polygonal windows (not for binary masks).
- translate Translation correction (Ohser, 1983). Implemented for all window geometries, but slow for complex windows.

Under CSR,  $K(r) = \pi r^2$ .

Under regularity,  $K(r)$  tends to be less than  $\pi r^2$ .

Under clustering,  $K(r)$  tends to be greater than  $\pi r^2$ .

Therefore, the plot of  $K(r)$  indicates that the longleaf pines are clustered.

## 6. Appendix

```
##### 1. Data Description

library(spatstat)
data(longleaf)
-----

plot(longleaf)
plot(unmark(longleaf))
-----

longleaf$x
longleaf$y
longleaf$marks

##### 3.1 Quadrat counts

n<-200
qX <- quadratcount(longleaf, n, n)
plot(unmark(longleaf), pch="+")
plot(qX, add=TRUE, col="red", cex=0.6, lty=1)
(sum(qX>0)/n^2)/((200/n)^2)

lambda<-function(n){
qX <- quadratcount(longleaf, n, n)
(sum(qX>0)/n^2)/((200/n)^2)}
x<-c(20,50,100,200,400,800)
for(i in 1:6)
  y[i]<-lambda(x[i])
plot(x,y,type="l")

##### 3.2 Kernel estimation

par(mfrow=c(3,3))
for(i in 1:9) plot(density.ppp(longleaf, 3*i), main=3*i)

par(mfrow=c(1,1))
plot(density.ppp(longleaf, 15),main="Sigma=15.0")

##### 4.1 Quadrat count test for CSR

##100 random circular quadrats(not overlaped)##
rx<-matrix(300,100,2)
rx[1,1]<-runif(1,6,194)
rx[1,2]<-runif(1,6,194)
rx[2,1]<-runif(1,6,194)
rx[2,2]<-runif(1,6,194)

for (i in 1:100){
  while(sum(nndist(rx[1:i,])<12)>0 || rx[i,1]>200){
    rx[i,1]<-runif(1,6,194)
    rx[i,2]<-runif(1,6,194)
  }
}
circular<-rx
plot(rx,pch="+",col=2)
for(j in 1:100){
  plot(disc(6,rx[j,]),add=TRUE)
}
#####100 circular quadrats on longleaf data#####
plot(unmark(longleaf),pch="+")
for(j in 1:100){
  plot(disc(6,rx[j,]),add=TRUE)
}
##-----count the number of trees in each circle-----

count<-numeric(100)
for(i in 1:100){
  count[i]<-sum(sqrt((longleaf$x-rx[i,1])^2+(longleaf$y-rx[i,2])^2)<6)
}

```

```
##----- chi-square goodness of fit test-----
O<-numeric(11)
for(j in 1:11)
  O[j]<-sum(count==j-1)
E<-numeric(11)
for(j in 1:11)
  E[j]<-exp(-1.43)*(1.43)^(j-1)/(factorial(j-1))*100
```

```
T<-matrix(0,6,2)
T[1:5,1]<-O[1:5]
T[6,1]<-sum(O[6:11])
T[1:5,2]<-E[1:5]
T[6,2]<-sum(E[6:11])
pearson<-sum((T[,1]-T[,2])^2/T[,2])
pvalue<-1-pchisq(pearson,5)
pvalue
```

```
-----
T<-matrix(0,6,2)
T[,1]<-c(34,33,17,7,3,6)
T[,2]<-c(23.93,34.22,24.47,11.66,4.17,1.54)
pearson<-sum((T[,1]-T[,2])^2/T[,2])
pvalue<-1-pchisq(pearson,10)
pvalue
```

#### 4.2 Indices for Quadrat Count Data

```
-----
#####
count_mean<-mean(count)
count_var<-var(count)
I<-count_var/count_mean
ICS<-count_var/count_mean-1
ID<-sum((count-count_mean)^2)/100/count_mean
```

#### 5.1 G and F functions

```
#####
plot(Gest(longleaf),main="G(r)>F(r)")
plot(Fest(longleaf),add=TRUE,col="blue")
--
intensity<-584/40000
G<-function(r)
  1-exp(-intensity*pi*r^2)
plot(G,add=TRUE)
--
csr<-runifpoint(584, win=owin(c(0,200),c(0,200)))
plot(Gest(csr),add=TRUE)
```

#### 5.2 The K function

```
#####
plot(Kest(longleaf))
```