Quadrat and Distance Based Methods for Point Patterns

SERGIO REY

GPH 483/598
Geographic Information Analysis
School of Geographical Sciences
Arizona State University
Spring 2010

Outline

Quadrat Counts

Basic Approach

- Impose a tessellation over the area
- Count number of points in each cell
- Compare observed counts against expected counts under the null of CSR

Expected Counts

- Relies on relationship between Poisson-CSR-Binomial
- Treat each cell as independent
- $E[x_i] = \lambda |A_i|$ where λ is the overall area intensity and $|A_i|$ is the area of cell i

Quadrat Counts: Test Statistic

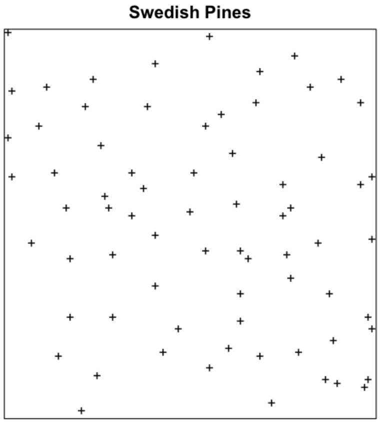
χ^2 statistic

- Regular tessellation (Grid with $m \times k$ cells)
- m rows
- n cols
- Equal sized cells

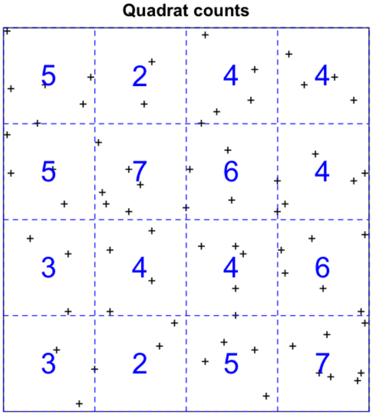
$$\chi^2 = \sum_{i=1}^{m} \sum_{j=1}^{k} (x_{i,j} - E[x_{i,j}])^2 / \lambda \quad (1)$$

Under the null of csr our test statistic has a $\chi^2(m \times k - 1)$ distribution

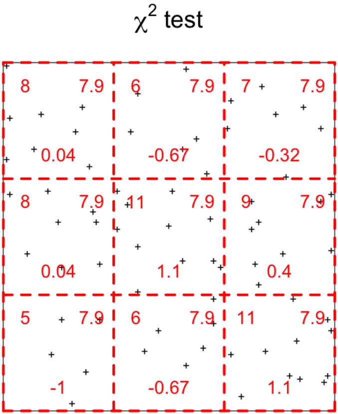
Quadrat Counts



Quadrat Counts



Quadrat Counts



$$\chi^2 = 4.6761,\ df = 8,\ p - value = 0.7916$$

Quadrat Counts

Issues

- Choice of tessellation
  - how many cells?
  - what cell shape?
  - locations random or fixed?
- Edge effects
- Spatial dependence
  - Independent cell counts
  - Independent locations

Monte Carlo Simulation

Basic Approach

- Specify test statistic
- Calculate test statistic on observed pattern: ψ
- Specify a null hypothesis (H_o)
- Specify an alternate hypothesis (H_1)
- Simulate Empirical Sampling Distribution of $\psi|H_o$
  - Draw *nsim* realizations under the null.
  - Calculate ψ_i where $i = 1, 2, \ldots, nsim$.
  - Compare ψ to distribution of ψ_i.

Computational Approximation to Inference

Motivations

- Substitute capital for labor
- Practical when no analytical results are available
- Very flexible

Issues

- Not generalizable beyond data at hand
- Less powerful than exact tests (if available)
- May be computationally expensive

Monte Carlo Simulation

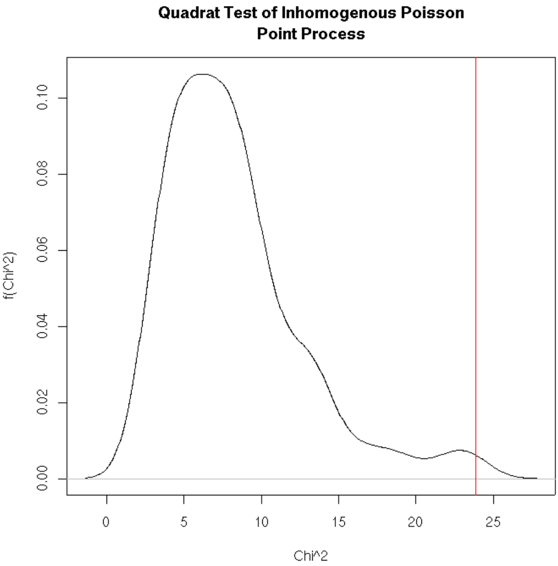
Basic Approach

- Specify test statistic
- Calculate test statistic on observed pattern: ψ
- Specify a null hypothesis (H_o)
- Specify an alternate hypothesis (H_1)
- Simulate Empirical Sampling Distribution of $\psi | H_o$
  - Draw *nsim* realizations under the null.
  - Calculate ψ_i where $i = 1, 2, \ldots, nsim$.
  - Compare ψ to distribution of ψ_i.

Code: ihhpsim.r

```
source("quadcounts.r")
source("ihppsim.r")
pp=ippsim(100)*9+1
ppt=quadcount(pp[,1],pp[,2])
set.seed(100)
nsim=99
source("hppsim.r")
results=matrix(0,nsim+1,1)
for(i in 1:nsim){
    pp=csr(100,1,1,10,10)
    t=quadcount(pp$x,pp$y)
    results[i]=t$chi2
}
results[100]=ppt$chi2
plot(density(results),main="Quadrat Test of Inhomogenous Po
Point Process",xlab="Chi^2",ylab="f(Chi^2)")
abline(v=ppt$chi2,col='red')
```

Empirical Sampling Distribution



Pseudo Significance Level

p-value

$$p(\chi^2) = \frac{1 + \sum_{i=1}^{nsim} \psi_i}{nsim + 1} \tag{2}$$

where:

$$\psi_i = \begin{cases} 1 & \text{if } \chi_i^2 \geq \chi^2, \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

p-value

$$p(\hat{\chi}^2) = \frac{1 + 0}{99 + 1} = 0.01 \tag{4}$$

Mean Nearest Neighbor Statistic

$d_{min}(s_i)$

$$d_{min}(s_i) = min(d_{i,1}, d_{i,2}, \ldots, d_{i,n}) \quad (5)$$

$d_{min}(s_i)$ is the distance between i and its nearest neighbor event.

Test Statistic

$$\bar{d}_{min} = \frac{1}{n}\sum_{i=1}^{n} d_{min}(s_i) \quad (6)$$

Originally suggested by Clark and Evans (1954)

Mean Nearest Neighbor Statistic Distribution

$\bar{d}_{min} \sim N(\mu, \sigma^2)$

$$\mu = E[\bar{d}_{min}] = 0.5(n^{-1}|A|)^{1/2} + (0.051 + 0.042n^{-1/2})n^{-1}P \qquad (7)$$

$$\sigma^2 = V[\bar{d}_{min}] = 0.070n^{-1/2}|A| + 0.037(n^{-5}|A|)^{1/2}P \qquad (8)$$

where $|A|$ and P are the area and perimeter of the study area, respectively.

Issues

- Approximation, not an exact result.
- Dependence of nearest neighbor distances is ignored.
- Distribution of $d_{min}(s_i)$ ignored (only first moment).

Edge Effects

Problems

- For points close the the boundary intensity is underestimated.
- Neighboring points are outside the study region.

Solutions

- Buffer the points
- Edge corrections
- Monte Carlo Simulations

Nearest Neighbor G Function

$G(d)$

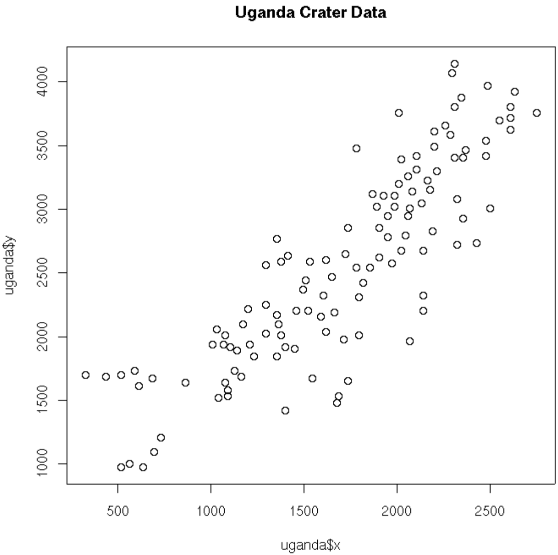
$$G(d) = \sum_{i=1}^{n} \Phi_i^d / n \tag{9}$$

where

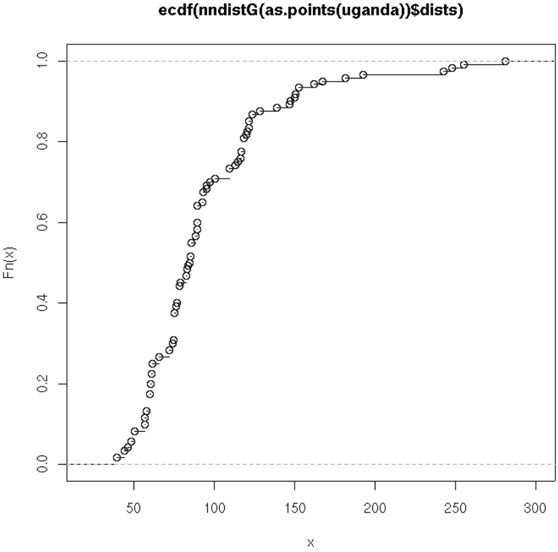
$$\Phi_i^d = \begin{cases} 1 & \text{if } d_{min}(s_i) < d \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

$G(d)$ is the proportion of nearest neighbor distances that are less than d.

Uganda Crater Data



Nearest Neighbor G Function



G Function Intepretation

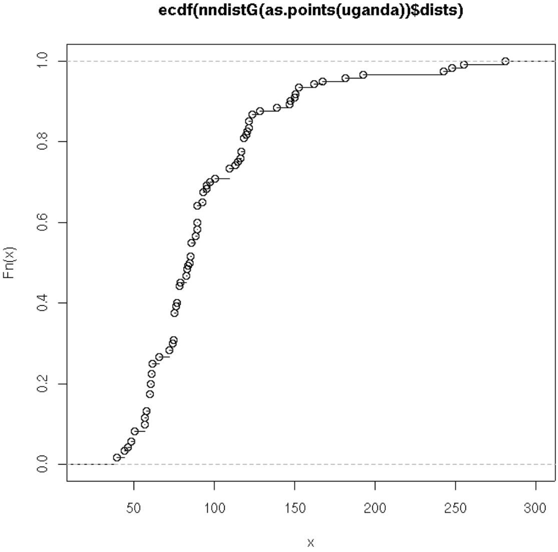
Shape

- G increasing rapidly at small distances points to *clustering*.
- G increases slowly points to *uniformity*.
- Both are deviations from CSR.

Compare G to that from a CSR Process

- Theoretical G
- Homogeneous Poisson process
- Density equal to density of actual pattern
- Empirical distribution against theoretical distribution
  - Should be a 45 degree line if process is CSR
  - Above the line = clustering
  - Below the line = dispersion

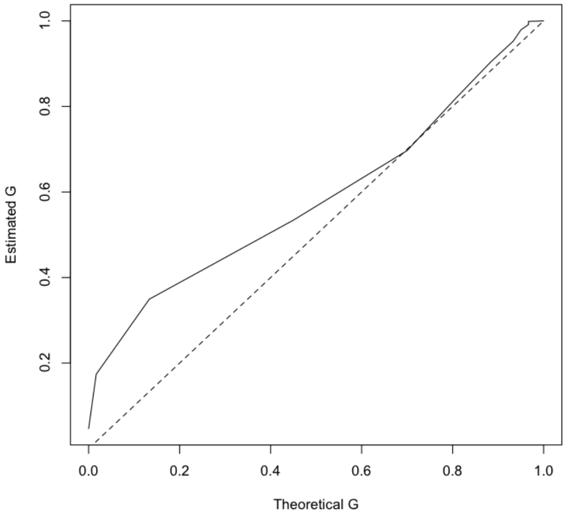
Nearest Neighbor G Function



Estimated vs. Theoretical G Function: Code

```
> library(splancs)
> data(uganda)
> plot(Ghat(as.points(uganda), seq(20, 500, 20)),
+ Fzero(pdense(as.points(uganda),  uganda$poly),
+ seq(20, 500, 20)), type="l",
+ xlab="Theoretical G",
+ ylab="Estimated G")
> lines(c(0,1),c(0,1),lty=2)
```

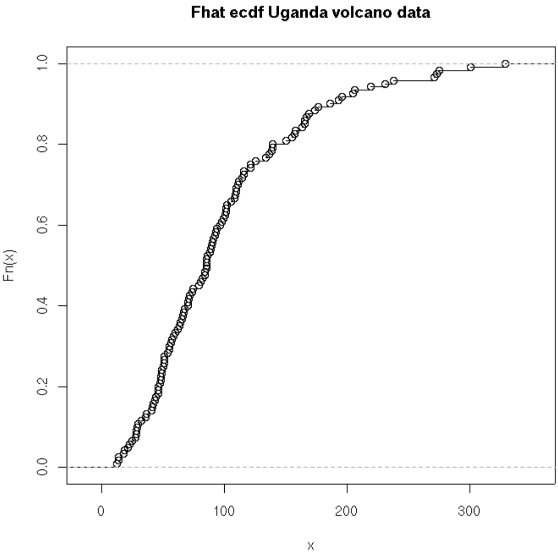
Estimated vs. Theoretical G Function



Nearest Neighbor F Function

- G function is sensitive to n
  - Can be rough
  - Takes on stepped appearance for small n
- Alternative approach is to generate N random points in the domain
  - Analyze the distribution of nearest event neighbor distances
  - Closest event to each point.
- Can be used for small n data sets

Nearest Neighbor F Function

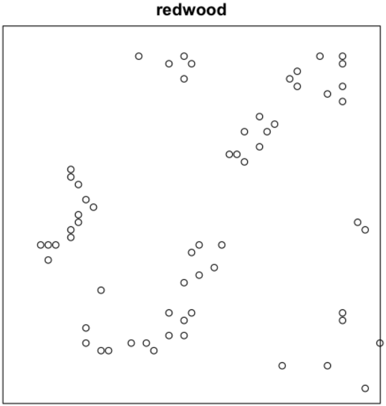


Nearest Neighbor J Function

$$J(d) = (1 - G(d))/(1 - F(d)) \tag{11}$$

- $J(d) < 1$ points to spatial clustering
- $J(d) > 1$ points to spatial regularity

Redwood

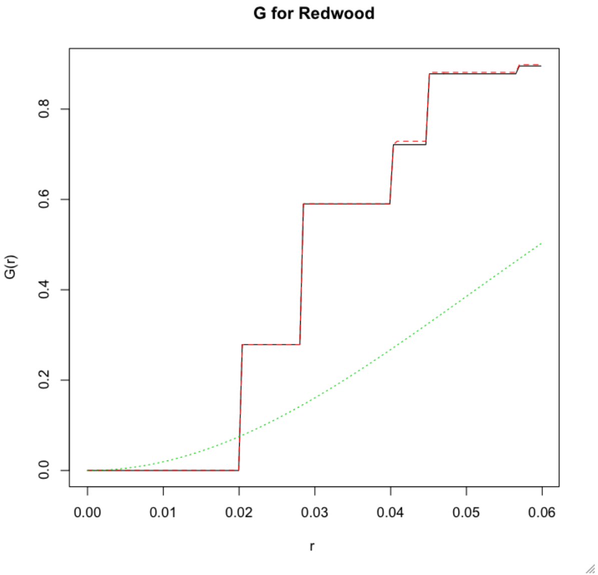


R code

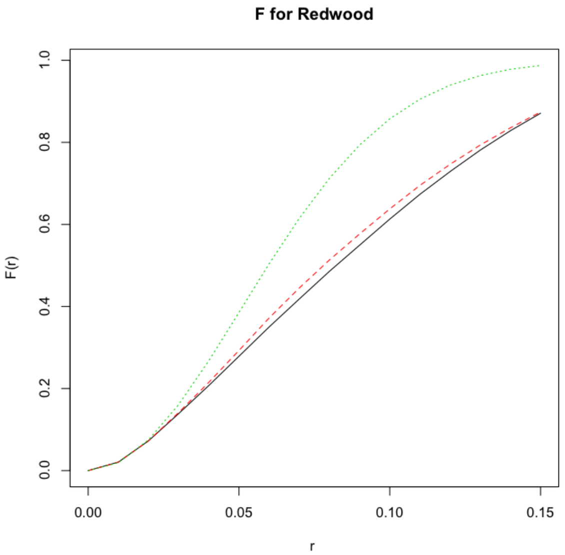
```
> gr=Gest(redwood)
> plot(gr,main="G for Redwood")
     lty col
km     1   1
rs     2   2
theo   3   3
```

- *km*: spatial Kaplan-Meier estimator of $G(r)$
- *rs*: the reduced sample edge correction estimator of $G(r)$
- *theo*: the theoretical value of $G(r)$ for a CSR process

Nearest Neighbor G Function



Nearest Neighbor F Function

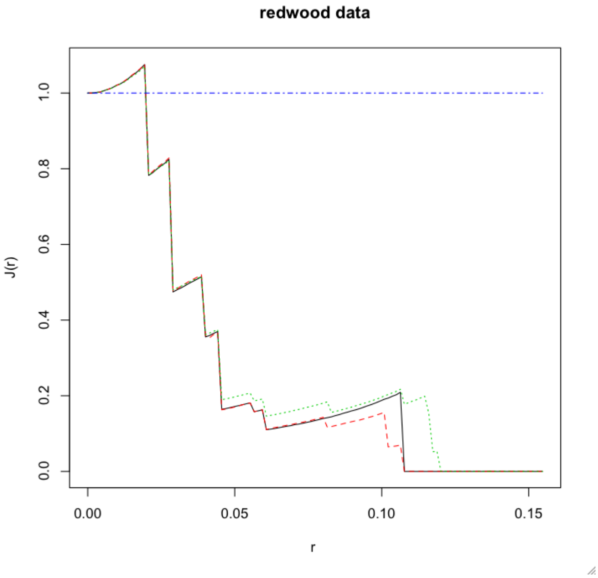


R code: J Function

```
>    J <- Jest(redwood, 0.01)
>    plot(J, main="redwood data")
     lty col
km     1   1
rs     2   2
un     3   3
theo   4   4
>    # values are below J= 1, indicating clustered pattern
```

- *km*: spatial Kaplan-Meier estimator of $G(r)$
- *rs*: the reduced sample edge correction estimator of $G(r)$
- *un*: the uncorrected estimate of $J(r)$ computed from the uncorrected estimates of F and G
- *theo*: the theoretical value of $J(r)$ for a CSR process

Nearest Neighbor J Function



Inter-Event Distance Distributions

G, F, and J Functions

- Take account of the nearest neighbor distributions: n distances or pieces of information
- Do not account for the full distribution of inter-event distances $n(n-1)/2$ distances.

Inter-Event Distances

- Consider all inter-event distances
- More than one distance for each point
- Second order analysis
  - Expresses the dependence of events
  - Spatial interaction

Ripley's K function

K

$$K(d) = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} \psi_{ij}(d)}{n\lambda} \tag{12}$$

where:

$$\psi_{ij}(d) = \begin{cases} 1 & \text{if } d_{ij} \leq d \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

Circle centered on each point s_i

$$\sum_{j=1}^{n} \psi_{ij}(d) \tag{14}$$

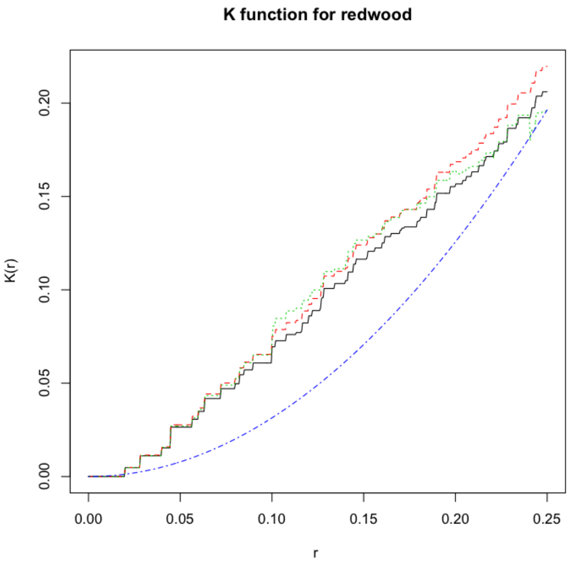
is the number of events within a circle of radius d centered on even s_i.

R code: K Function

```
> rk=Kest(redwood)
> plot(rk,main="K function for redwood")
       lty col
iso      1   1
trans    2   2
border   3   3
theo     4   4
```

- *iso*: Ripley's isotropic correction.
- *trans*: Translation correction.
- *border*: reduced sample estimator.
- *theo*: the theoretical value of K

K function



L function

Scaling of K

$$L(d) = \sqrt{K(d)/\pi} - d \tag{15}$$

Useful since:

$$E[K(d)] = \frac{\pi \lambda d^2}{\lambda} \tag{16}$$

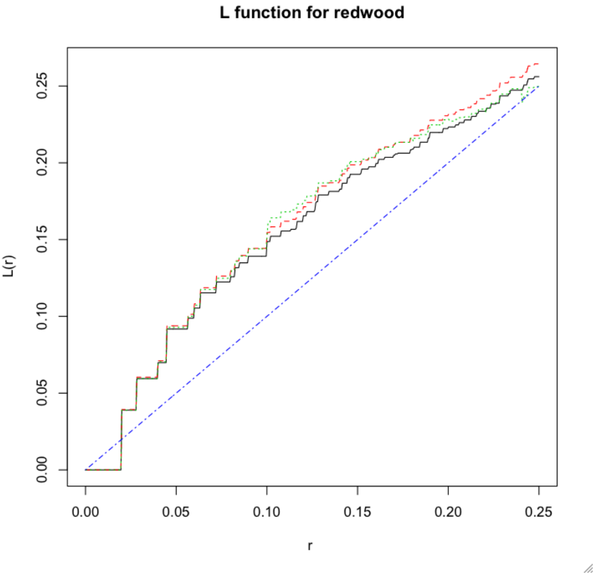
which can get large with d^2 and obscures small differences between expected and observed values.

R code: L Function

```
>  plot(rk, sqrt(./pi) ~ r, ylab="L(r)",
    + main="L function for redwood")
       lty col
iso      1   1
trans    2   2
border   3   3
theo     4   4
>
```

- *iso*: Ripley's isotropic correction.
- *trans*: Translation correction.
- *border*: reduced sample estimator.
- *theo*: the theoretical value of K

L function

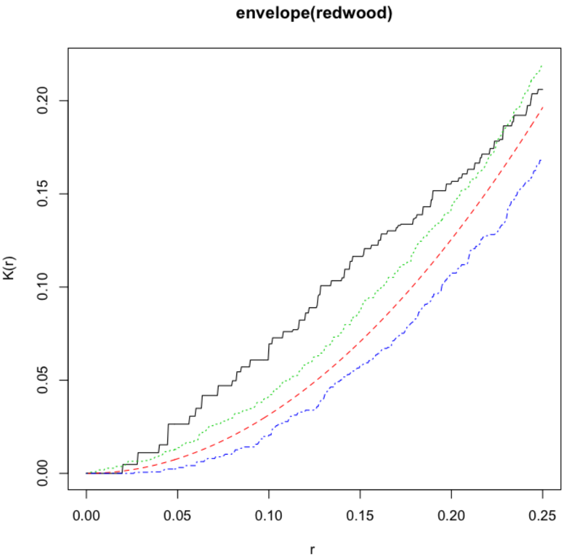


Simulation envelopes for *K*

```
> plot(envelope(redwood))
Generating 99 simulations of CSR ...
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, 29, 30,
31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45,
46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60,
61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75,
76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90,
91, 92, 93, 94, 95, 96, 97, 98, 99.

Done.
     lty col
obs    1   1
theo   2   2
hi     3   3
lo     4   4
```

K function simulation

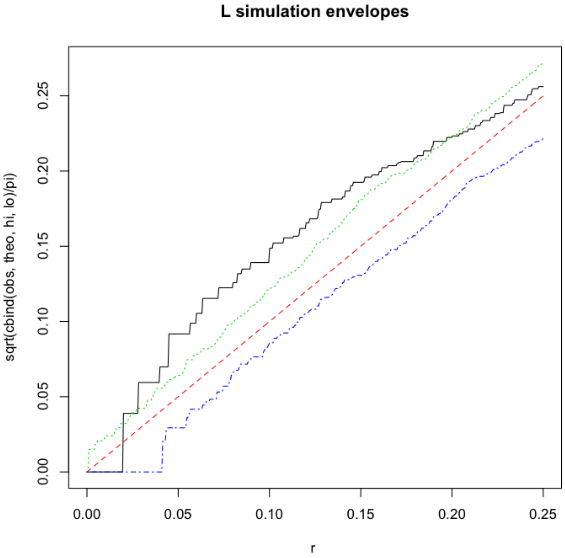


Simulation envelopes for L

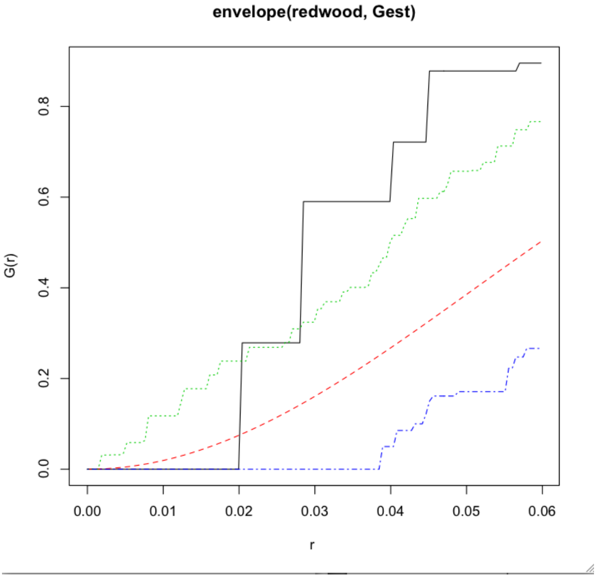
```
> E=envelope(redwood,Kest)
Generating 99 simulations of CSR ...
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,
16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30,
31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45,
46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60,
61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75,
76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90,
91, 92, 93, 94, 95, 96, 97, 98, 99.

Done.
> plot(E,sqrt(./pi)~r,main="L simulation envelopes")
     lty col
obs    1   1
theo   2   2
hi     3   3
lo     4   4
```

L function simulation



G function simulation



F function simulation