Homework 1

공간자료의 통계분석

2022-22742 진수정

```
library(tidyverse)
library(spatstat)
library(spdep)
library(sparr)
library(cubature)
library(plot3D)
```

Problem 1.

[20 pts] redwoodfull and amacrine are the datasets both available from spatstat package in R. For each dataset, use Z score of Clark and Evans (1954) (in the page 31 of the first lecture note) to determine whether the data has aggregating or repulsive tendency.

[Redwoodfull data]

```
n = redwoodfull$n
A = area(redwoodfull$window)

D = pairdist(redwoodfull)
diag(D) = max(D) # to avoid self-selected
di = apply(D,1,min) # nearest distance at each point

# Clark and Evans (1954) Z score
r0_bar = sum(di)/n
re = 1/(2*sqrt(n/A))
se = 0.261/sqrt(n^2/A)

Z = (r0_bar-re)/se
pval = 2 * (1 - pnorm(abs(Z),0,1))

cat("Z =", Z, "\n",
    "p-value (two tail) =",pval,"\n")
```

```
## Z = -5.521471
##  p-value (two tail) = 3.361737e-08
```

p-value < 0.05 이므로 유의수준 0.05 하에서 spatial uniformity가 성립하지 않는다고 볼 수 있다.

```r
cat("r0_bar / re =", r0_bar/re, "\n")
```

```
## r0_bar / re = 0.7936008
```

또한 $\bar{r_0}/r_e$ 가 1보다 작으므로 aggregating tendency가 있다고 볼 수 있다.

[amacrine data]

```r
n = amacrine$n
A = area(amacrine$window)

D = pairdist(amacrine)
diag(D) = max(D) # to avoid self-selected
di = apply(D,1,min) # nearest distance at each point

# Clark and Evans (1954) Z score
r0_bar = sum(di)/n
re = 1/(2*sqrt(n/A))
se = 0.261/sqrt(n^2/A)

Z = (r0_bar-re)/se
pval = 2 * (1 - pnorm(abs(Z),0,1))

cat("Z =", Z, "\n",
    "p-value (two tail) =",pval,"\n")
```

```
## Z = 5.779404
##  p-value (two tail) = 7.496553e-09
```

p-value < 0.05 이므로 유의수준 0.05 하에서 spatial uniformity가 성립하지 않는다고 볼 수 있다.

```r
cat("r0_bar / re =", r0_bar/re, "\n")
```

```
## r0_bar / re = 1.175946
```

또한 $\bar{r_0}/r_e$ 가 1보다 크므로 repulsive tendency가 있다고 볼 수 있다.

Problem 2.

[20 pts] The dataset lansing (available in spatstat package in R) provides locations of trees in Lansing Woods, Clinton County, Michigan, USA by D.J. Gerrard. The original plot size (924 x 924 ft) is rescaled to the unit square. The data contain six classification (hickories, maples, red oaks, white oaks, black oaks and miscellaneous trees). For each category of tree type, apply two of quadrat methods with an appropriate quadrat size.

```
hickory = split(lansing)$hickory
maple = split(lansing)$maple
redoak = split(lansing)$redoak
whiteoak = split(lansing)$whiteoak
blackoak = split(lansing)$blackoak
misc = split(lansing)$misc
```

▶ Unconditional approach (chi-square test)

```
quadrat_X2 = function(data,nx,ny,k,alpha) {
  n = data$n
  m = nx * ny

  # Observed counts in each cell
  obs_by_cell = quadratcount(data,nx,ny)
  Oi = tabulate(match(obs_by_cell,0:max(obs_by_cell)))
  cat("Observed counts\n",Oi,"\n")
  Oi = c(Oi[1:(k-1)],sum(Oi[k:max(obs_by_cell)]))

  # Expected counts in each cell
  expd_rate = n / m
  expd_prob = c(dpois(0:(k-2),expd_rate),
                1-sum(dpois(0:(k-2),expd_rate)))
  Ei = m * expd_prob
  cat("Expected counts\n",round(Ei),"\n\n")

  # X2 statistics
  X2 = sum((Oi-Ei)^2/Ei)
  pval = 1 - pchisq(X2,df = k-1)

  if (pval < alpha) {
    cat("X2:",X2,"\n",
        "p-value:",pval,"\n",
        "=> Reject uniformity null")
  } else {
    cat("X2:",X2,"\n",
        "p-value:",pval,"\n",
        "=> Cannot reject uniformity null")
  }
}
```

▶ Conditional approach (VMR test)

```r
quadrat_VMR = function(data,nx,ny,alpha) {
  n = data$n
  m = nx * ny

  Oi = quadratcount(data,nx,ny)
  O_bar = n / m

  # variance-mean ratio (VMR)
  VMR = sum((Oi-O_bar)^2)/((m-1)*O_bar)
  pval = 1 - pchisq((m-1)*VMR,df = m-1)

  if (pval < alpha) {
    cat("VMR:",VMR,"\n",
        "p-value:",pval,"\n",
        "=> Reject uniformity null")
  } else {
    cat("VMR:",VMR,"\n",
        "p-value:",pval,"\n",
        "=> Cannot reject uniformity null")
  }
}
```

Quadrat method 에서 $(expected\ frequency\ per\ cells) >= 5$ 가 되도록 적절한 quadrat size를 선택하였다.

[Hickory data]

1. Result of chi-square test

```r
quadrat_X2(hickory,9,9,10,0.05)
```

```
## Observed counts
##  1 6 6 5 6 6 5 4 7 3 4 2 3 5 3 5 1 3 2 0 0 1 1 1 0 0 1
## Expected counts
##  0 0 1 2 3 6 8 10 11 41
##
## X2: 435.5788
##  p-value: 0
##  => Reject uniformity null
```

2. Result of VMR test

```r
quadrat_VMR(hickory,9,9,0.05)
```

```
## VMR: 4.061024
##  p-value: 0
##  => Reject uniformity null
```

[Maple data]

1. Result of chi-square test

```
quadrat_X2(maple,7,9,10,0.05)
```

```
## Observed counts
##  6 8 6 3 3 0 2 4 3 1 2 2 2 6 6 2 0 0 2 1 0 2 1 1
## Expected counts
##  0 0 1 2 3 5 7 9 9 27
##
## X2: 2469.009
##  p-value: 0
##  => Reject uniformity null
```

2. Result of VMR test

```
quadrat_VMR(maple,7,9,0.05)
```

```
## VMR: 5.393812
##  p-value: 0
##  => Reject uniformity null
```

[Redoak data]

1. Result of chi-square test

```
quadrat_X2(redoak,7,7,8,0.05)
```

```
## Observed counts
##  1 2 1 2 7 6 8 2 4 3 2 4 2 3 1 0 1
## Expected counts
##  0 0 1 2 4 6 7 27
##
## X2: 34.89223
##  p-value: 1.171826e-05
##  => Reject uniformity null
```

2. Result of VMR test

```
quadrat_VMR(redoak,7,7,0.05)
```

```
## VMR: 1.943762
##  p-value: 9.793053e-05
##  => Reject uniformity null
```

[Whiteoak data]

1. Result of chi-square test

```
quadrat_X2(whiteoak,9,7,11,0.05)
```

```
## Observed counts
##  0 2 2 6 4 11 7 6 4 4 8 2 1 3 1 1 0 0 0 1
## Expected counts
##  0 0 1 3 5 8 9 9 8 7 11
##
## X2: 19.10895
##  p-value: 0.03890043
##  => Reject uniformity null
```

2. Result of VMR test

```
quadrat_VMR(whiteoak,9,7,0.05)
```

```
## VMR: 1.855847
##  p-value: 4.933326e-05
##  => Reject uniformity null
```

[Blackoak data]

1. Result of chi-square test

```
quadrat_X2(blackoak,4,5,7,0.05)
```

```
## Observed counts
##  1 3 2 4 0 1 2 0 0 1 0 1 0 1 2 0 1 0 0 0 0 0 1
## Expected counts
##  0 0 1 1 2 3 13
##
## X2: 107.6366
##  p-value: 0
##  => Reject uniformity null
```

2. Result of VMR test

```
quadrat_VMR(blackoak,4,5,0.05)
```

```
## VMR: 5.736842
##  p-value: 1.221245e-14
##  => Reject uniformity null
```

[misc data]

1. Result of chi-square test

```r
quadrat_X2(misc,3,5,8,0.05)
```

```
## Observed counts
##  3 1 0 2 0 1 0 3 0 0 0 2 0 0 0 1 1 0 0 1
## Expected counts
##  0 0 0 1 1 2 2 8
##
## X2: 666.9978
##  p-value: 0
##  => Reject uniformity null
```

2. Result of VMR test

```r
quadrat_VMR(misc,3,5,0.05)
```

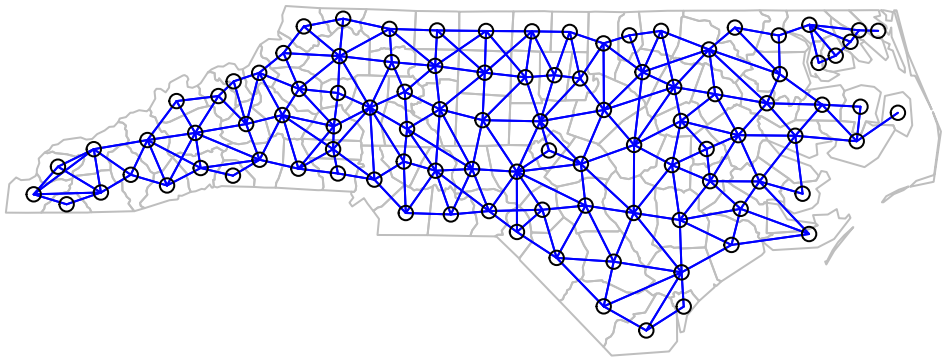
```
## VMR: 5.510204
##  p-value: 9.550849e-11
##  => Reject uniformity null
```

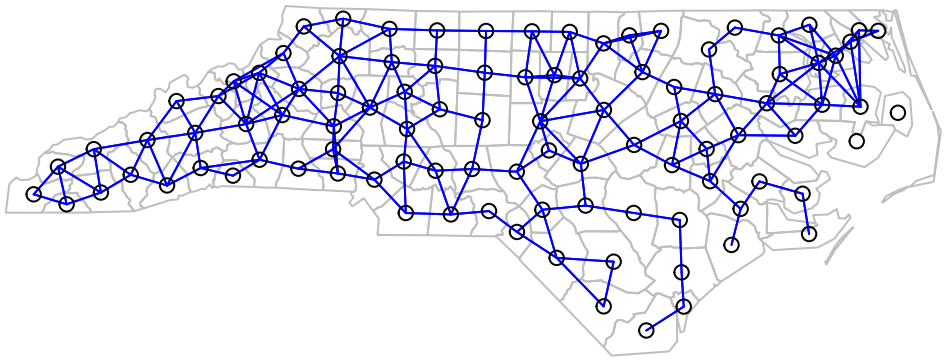
Problem 3.

[20 pts] The dataset nc.sids (available in spdep package in R) contains the data on sudden infant deaths in North Carolina for 1974-78 and 1979-84. The data set also contains the neighbour list given by Cressie and Chan (1989) omitting self-neighbours (ncCC89.nb), and the neighbour list given by Cressie and Read (1985) for contiguities (ncCR85.nb). Use example(nc.sids) to read the data set from shapefile, together with import of two different list of neighbours. Using the contiguous neighbor structure (ncCR85.nb), run Moran's I and Geary's C for SID deaths during 1979-84 (SID79).

```
example(nc.sids)
```

```
##
## nc.sds> if (requireNamespace("rgdal", quietly = TRUE)) {
## nc.sds+   library(rgdal)
## nc.sds+   if (requireNamespace("spdep", quietly = TRUE)) {
## nc.sds+     library(spdep)
## nc.sds+     nc.sids <- readOGR(system.file("shapes/sids.shp", package="spData")[1])
## nc.sds+     proj4string(nc.sids) <- CRS("+proj=longlat +ellps=clrk66")
## nc.sds+     row.names(nc.sids) <- as.character(nc.sids$FIPS)
## nc.sds+     rn <- row.names(nc.sids)
## nc.sds+     ncCC89_nb <- read.gal(system.file("weights/ncCC89.gal", package="spData")[1],
## nc.sds+                           region.id=rn)
## nc.sds+     ncCR85_nb <- read.gal(system.file("weights/ncCR85.gal", package="spData")[1],
## nc.sds+                           region.id=rn)
## nc.sds+
## nc.sds+     plot(nc.sids, border="grey")
## nc.sds+     plot(ncCR85_nb, coordinates(nc.sids), add=TRUE, col="blue")
## nc.sds+     plot(nc.sids, border="grey")
## nc.sds+     plot(ncCC89_nb, coordinates(nc.sids), add=TRUE, col="blue")
## nc.sds+   }
## nc.sds+ }
## OGR data source with driver: ESRI Shapefile
## Source: "C:\Program Files\R\R-4.0.3\library\spData\shapes\sids.shp", layer: "sids"
## with 100 features
## It has 22 fields
```





▶ Moran's I

```
ncCR85 = nb2listw(ncCR85.nb)
ncCR85$n
```

```
## Neighbour list object:
## Number of regions: 100
## Number of nonzero links: 492
## Percentage nonzero weights: 4.92
## Average number of links: 4.92
```

지역의 개수가 충분히 크므로 (m = 100 > 30), I의 분포를 정규분포로 근사해서 moran's I test를 진행하였다.

```
moran.test(nc.sids$SID79, ncCR85, randomisation = F)
```

```
##
##  Moran I test under normality
##
## data:  nc.sids$SID79
## weights: ncCR85
##
## Moran I statistic standard deviate = 2.4323, p-value = 0.007503
## alternative hypothesis: greater
## sample estimates:
```

```
## Moran I statistic       Expectation          Variance
##       0.149827692      -0.010101010       0.004323492
```

▶ Geary's C

```
geary.test(nc.sids$SID79, ncCR85, randomisation = F)
```

```
##
##  Geary C test under normality
##
## data:  nc.sids$SID79
## weights: ncCR85
##
## Geary C statistic standard deviate = 2.1616, p-value = 0.01532
## alternative hypothesis: Expectation greater than statistic
## sample estimates:
## Geary C statistic       Expectation          Variance
##       0.849098534       1.000000000       0.004873344
```

Moran's I 와 Geary's C 결과를 통해 데이터에 spatial dependence 가 있다고 볼 수 있다.

Problem 4.

[20 pts] Create EDF plots for ˆH (t) v.s. H(t) and ˆG(y) v.s. G(y) with envelopes by your own without using built-in functions. Discuss your findings. For the envelopes, use s = 100.

▶ CSR test based on inter-event distance

```
# H(t): theoretical CDF
CDF_H = function(t, area_type) {
  if (area_type == 'square') {
    if (t >= 0 & t <= 1) {return (pi*t^2-8/3*t^3+1/2*t^4)}
    else if (t > 1 & t <= sqrt(2)) {return (1/3-2*t^2-1/2*t^4+4/3*(t^2-1)^(1/2)*(2*t^2+1)+2*t^2*asin(2*
    else if (t < 0) {return (0)}
    else {return (1)}
  } else if (area_type == 'circle') {
    if (t >= 0 & t <= 2) {
    return (1+1/pi*(2*(t^2-1)*acos(t/2)-t*(1+t^2/2)*sqrt(1-t^2/4)))}
    else if (t < 0) {return (0)}
    else {return (1)}
  }
}
CDF_H = Vectorize(CDF_H, vectorize.args = 't')

# H_hat(t): empirical CDF
ECDF_H = function(data,t) {
  tij = pairdist(data)
  return (mean(tij <= t))
}
ECDF_H = Vectorize(ECDF_H, vectorize.args = 't')
```

▶ CSR test based on nearest neighbor distance

```
# G(y): theoretical CDF
CDF_G = function(data,y) {
  n = data$n
  A = area(data$window)
  lambda = n*A^(-1)
  return ((1-exp(-lambda*pi*y^2)) * (y>=0))
}
CDF_G = Vectorize(CDF_G, vectorize.args = 'y')

# G_hat(y): empirical CDF
ECDF_G = function(data,y) {
  D = pairdist(data)
  diag(D) = max(D)
  yi = apply(D,1,min)
  return (mean(yi <= y))
}
ECDF_G = Vectorize(ECDF_G, vectorize.args = 'y')
```

```
# G(y): theoretical CDF
CDF_G = function(data,y) {
  n = data$n
```

```
  A = diff(data$window$xrange) * diff(data$window$yrange)
  lambda = n*A^(-1)
  return ((1-exp(-lambda*pi*y^2)) * (y>=0))
}
CDF_G = Vectorize(CDF_G, vectorize.args = 'y')

# G_hat(y): empirical CDF
ECDF_G = function(data,y) {
  dat = matrix(c(data$x,data$y),nrow = data$n,ncol = 2)
  D = as.matrix(dist(dat,method = 'euclidean'))
  diag(D) = max(D)
  yi = apply(D,1,min)
  return (mean(yi <= y))
}
ECDF_G = Vectorize(ECDF_G, vectorize.args = 'y')
```

[japanesepines data]

```
# area type: square
japanesepines$window
```

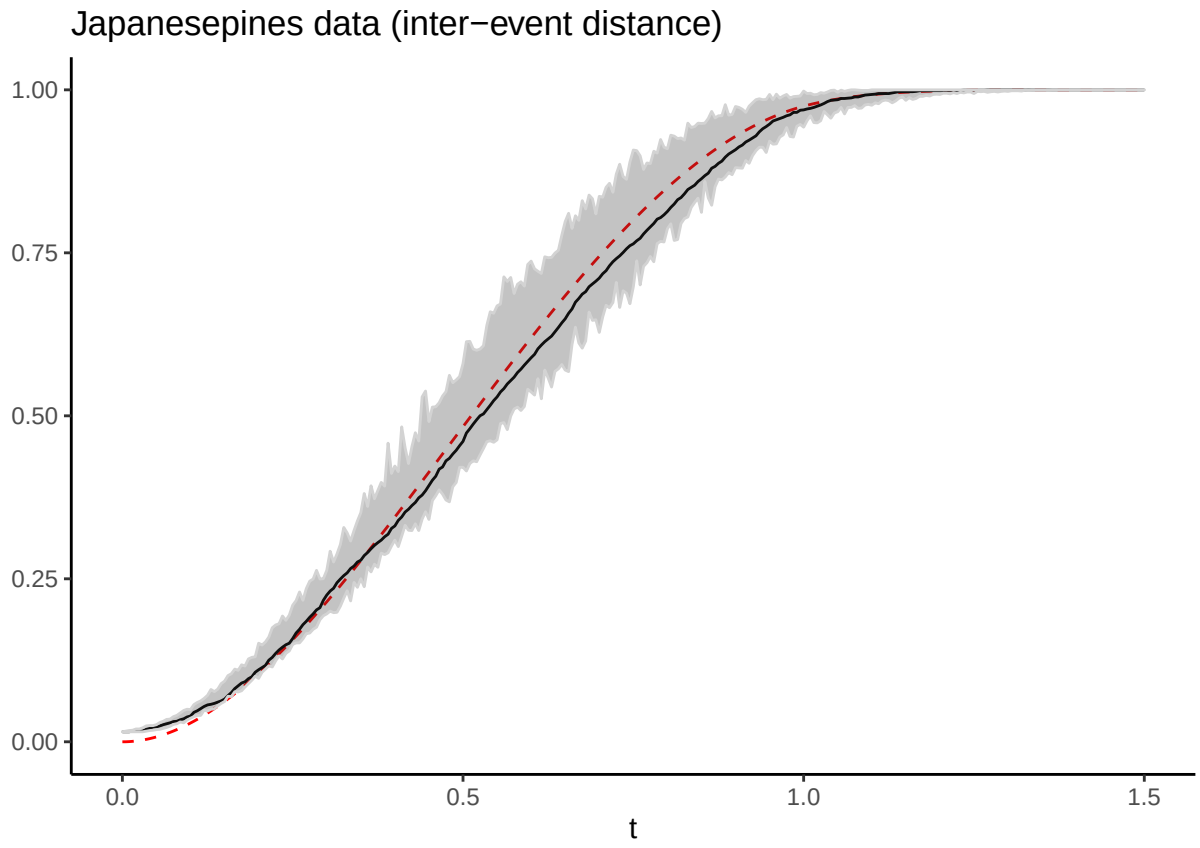
```
## window: rectangle = [0, 1] x [0, 1] units (one unit = 5.7 metres)
```

- $H(t)$ *vs.* $\hat{H}(t)$

```
res.j = data.frame(
  t = seq(0,1.5,0.005),
  U = NA,
  L = NA
)

res.j$H = CDF_H(res.j$t,'square')
res.j$H_hat = ECDF_H(japanesepines,res.j$t)

# envelop
n = japanesepines$n
A = owin(japanesepines$window$xrange, japanesepines$window$yrange)
s = 100
set.seed(42)

for (j in 1:nrow(res.j)) {
  t = res.j[j,'t']
  Hi_hat = rep(NA,s-1)

  for (i in 1:(s-1)) {
    dat_null = runifpoint(n,win = A)
    Hi_hat[i] = ECDF_H(dat_null,t)
  }
```

```
  res.j[j,'U'] = max(Hi_hat)
  res.j[j,'L'] = min(Hi_hat)
}
```

```
res.j %>%
  ggplot() +
  geom_line(aes(x = t,y = H),col = 'red',linetype = 'dashed') +
  geom_line(aes(x = t,y = H_hat)) +
  geom_ribbon(aes(x = t,ymin = L,ymax = U),col = 'lightgray',alpha = 0.3) +
  theme_classic() +
  ggtitle("Japanesepines data (inter-event distance)") +
  ylab("")
```

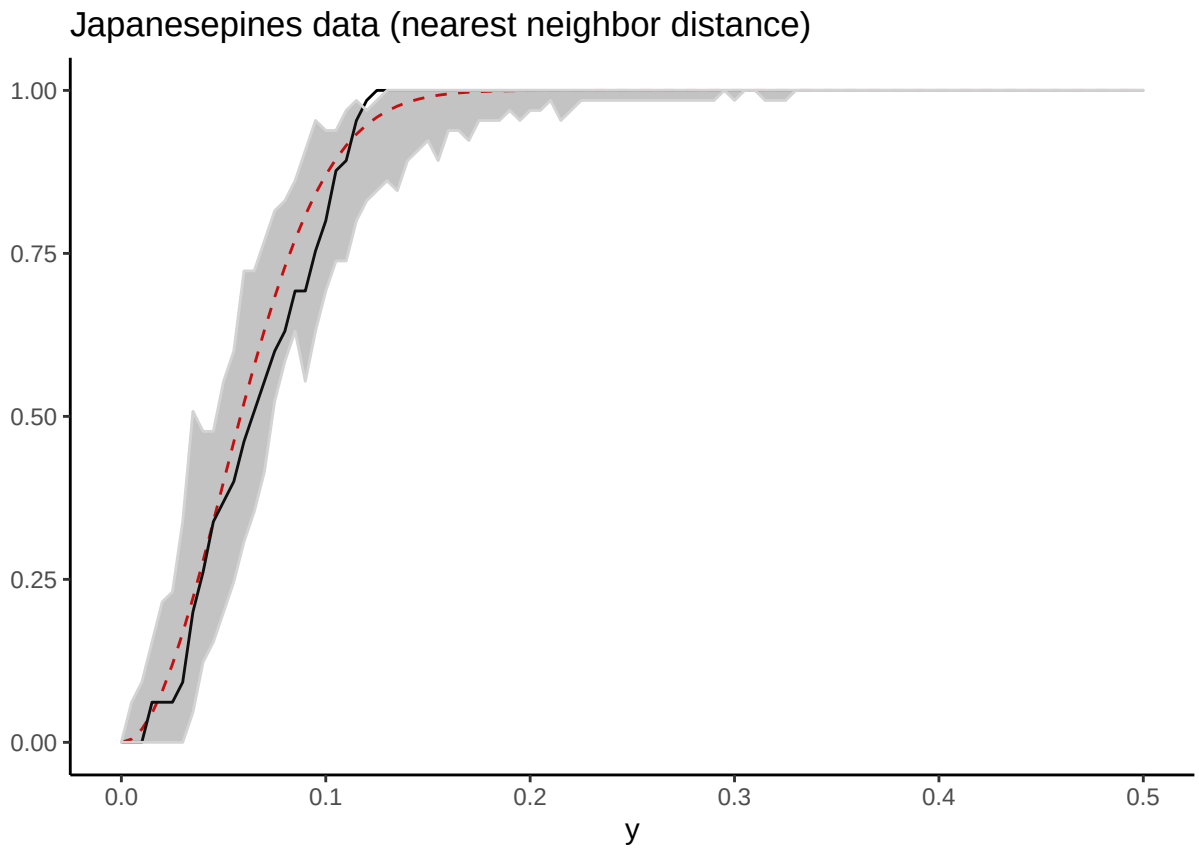


- $G(y)$ *vs.* $\hat{G}(y)$

```
res.j = data.frame(
  y = seq(0,0.5,0.005),
  U = NA,
  L = NA
)
res.j$G = CDF_G(japanesepines,res.j$y)
res.j$G_hat = ECDF_G(japanesepines,res.j$y)

# envelop
n = japanesepines$n
A = owin(japanesepines$window$xrange, japanesepines$window$yrange)
```

```r
s = 100
set.seed(42)

for (j in 1:nrow(res.j)) {
  y = res.j[j,'y']
  Gi_hat = rep(NA,s-1)

  for (i in 1:(s-1)) {
    dat_null = runifpoint(n,win = A)
    Gi_hat[i] = ECDF_G(dat_null,y)
  }

  res.j[j,'U'] = max(Gi_hat)
  res.j[j,'L'] = min(Gi_hat)
}
```

```r
res.j %>%
  ggplot() +
  geom_line(aes(x = y,y = G),col = 'red',linetype = 'dashed') +
  geom_line(aes(x = y,y = G_hat)) +
  geom_ribbon(aes(x = y,ymin = L,ymax = U),col = 'lightgray',alpha = 0.3) +
  theme_classic() +
  ggtitle("Japanesepines data (nearest neighbor distance)") +
  ylab("")
```



두 가지 방법들에서 데이터의 theoretical CDF와 empirical CDF가 크게 차이나지 않았으며, 전체적으로

envelop를 벗어나지 않았다. 따라서 CSR이 성립하지 않다고 볼 근거가 없다.

[redwood data]

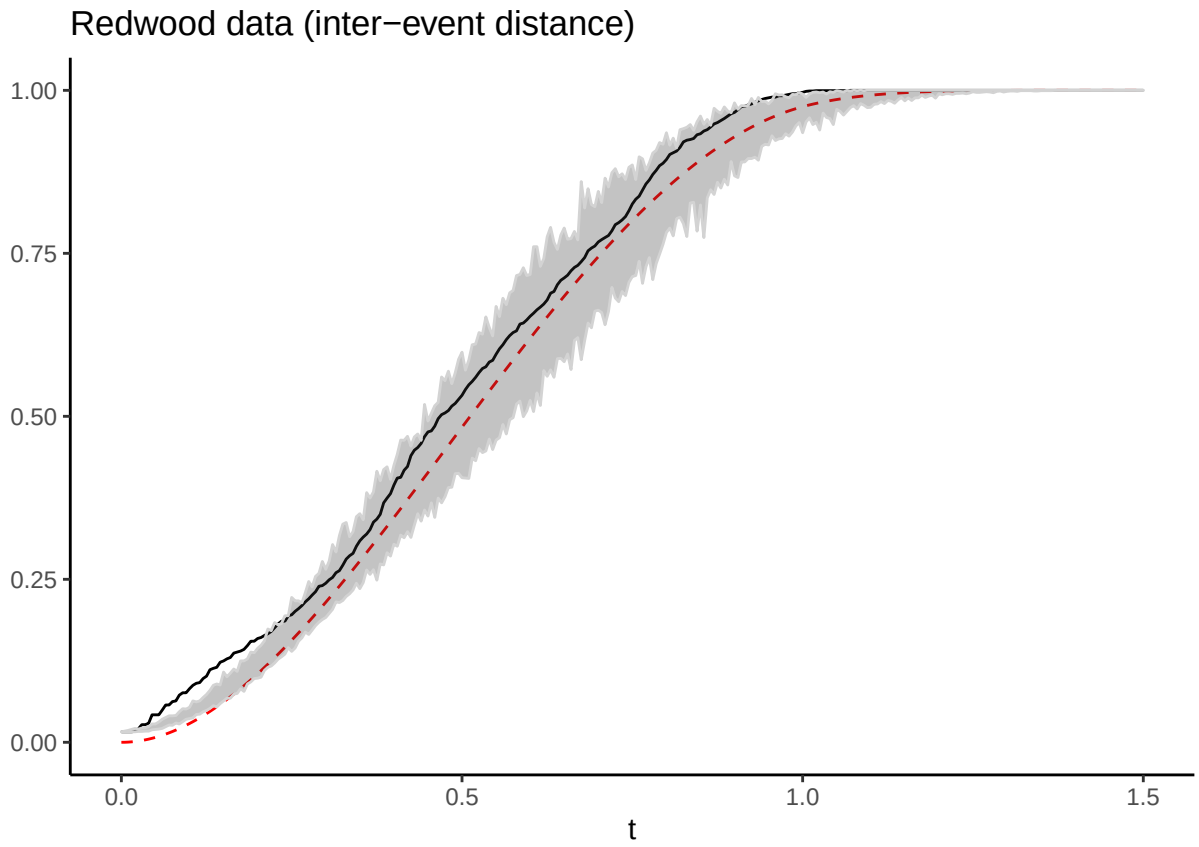
```
# area type: square
redwood$window
```

```
## window: rectangle = [0, 1] x [-1, 0] units
```

- $H(t)$ $vs.$ $\hat{H}(t)$

```
res.r = data.frame(
  t = seq(0,1.5,0.005),
  U = NA,
  L = NA
)

res.r$H = CDF_H(res.r$t,'square')
res.r$H_hat = ECDF_H(redwood,res.r$t)

# envelop
n = redwood$n
A = owin(redwood$window$xrange, redwood$window$yrange)
s = 100
set.seed(42)

for (j in 1:nrow(res.r)) {
  t = res.r[j,'t']
  Hi_hat = rep(NA,s-1)

  for (i in 1:(s-1)) {
    dat_null = runifpoint(n,win = A)
    Hi_hat[i] = ECDF_H(dat_null,t)
  }

  res.r[j,'U'] = max(Hi_hat)
  res.r[j,'L'] = min(Hi_hat)
}
```

```
res.r %>%
  ggplot() +
  geom_line(aes(x = t,y = H),col = 'red',linetype = 'dashed') +
  geom_line(aes(x = t,y = H_hat)) +
  geom_ribbon(aes(x = t,ymin = L,ymax = U),col = 'lightgray',alpha = 0.3) +
  theme_classic() +
  ggtitle("Redwood data (inter-event distance)") +
  ylab("")
```

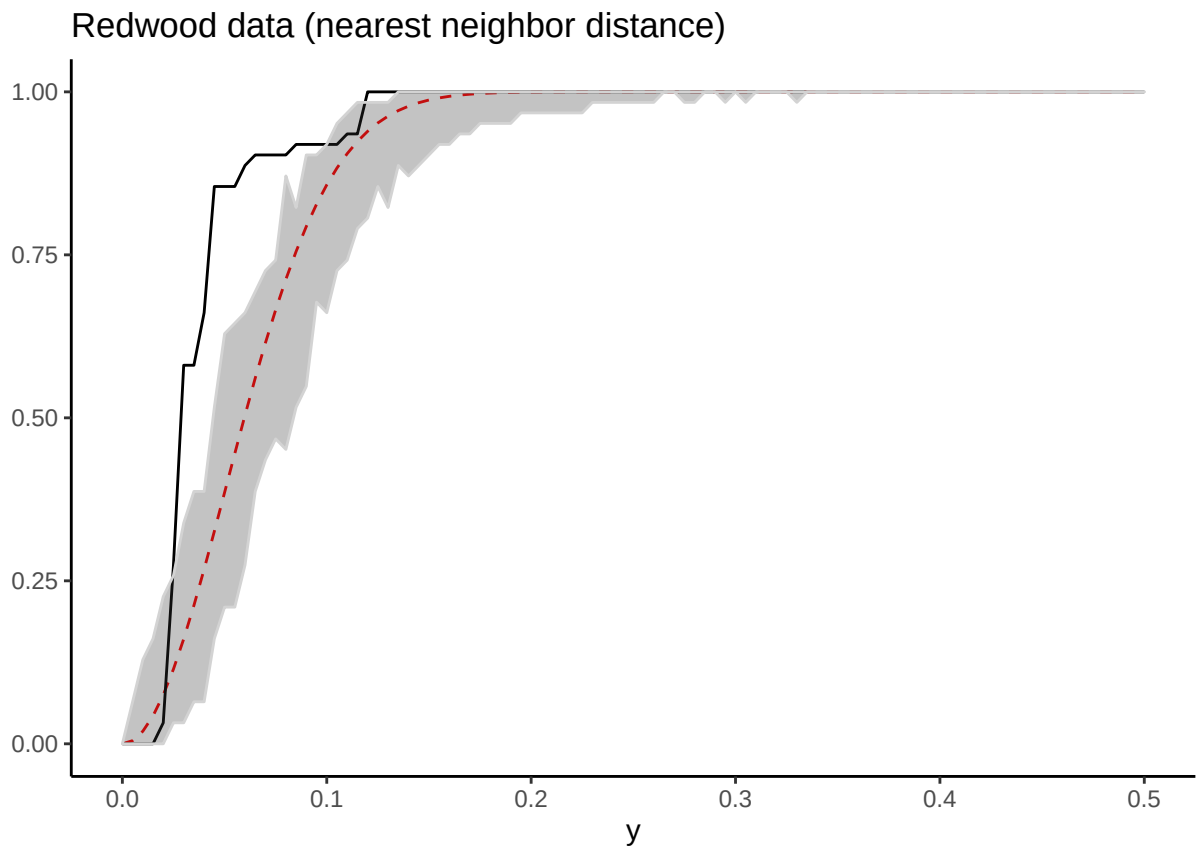


- $G(y)$ *vs.* $\hat{G}(y)$

```
res.r = data.frame(
  y = seq(0,0.5,0.005),
  U = NA,
  L = NA
)
res.r$G = CDF_G(redwood,res.r$y)
res.r$G_hat = ECDF_G(redwood,res.r$y)

# envelop
n = redwood$n
A = owin(redwood$window$xrange, redwood$window$yrange)
s = 100
set.seed(42)

for (j in 1:nrow(res.r)) {
  y = res.r[j,'y']
  Gi_hat = rep(NA,s-1)

  for (i in 1:(s-1)) {
    dat_null = runifpoint(n,win = A)
    Gi_hat[i] = ECDF_G(dat_null,y)
  }

  res.r[j,'U'] = max(Gi_hat)
```

```
  res.r[j,'L'] = min(Gi_hat)
}
```

```
res.r %>%
  ggplot() +
  geom_line(aes(x = y,y = G),col = 'red',linetype = 'dashed') +
  geom_line(aes(x = y,y = G_hat)) +
  geom_ribbon(aes(x = y,ymin = L,ymax = U),col = 'lightgray',alpha = 0.3) +
  theme_classic() +
  ggtitle("Redwood data (nearest neighbor distance)") +
  ylab("")
```



두 가지 방법들에서 작은 t값에 대해 empirical CDF가 envelop를 벗어나 위쪽에 위치해 있는 것을 확인할 수 있다. 이를 통해 redwood data에 aggregating tendency가 있을 것이라 생각할 수 있다.

[cells data]

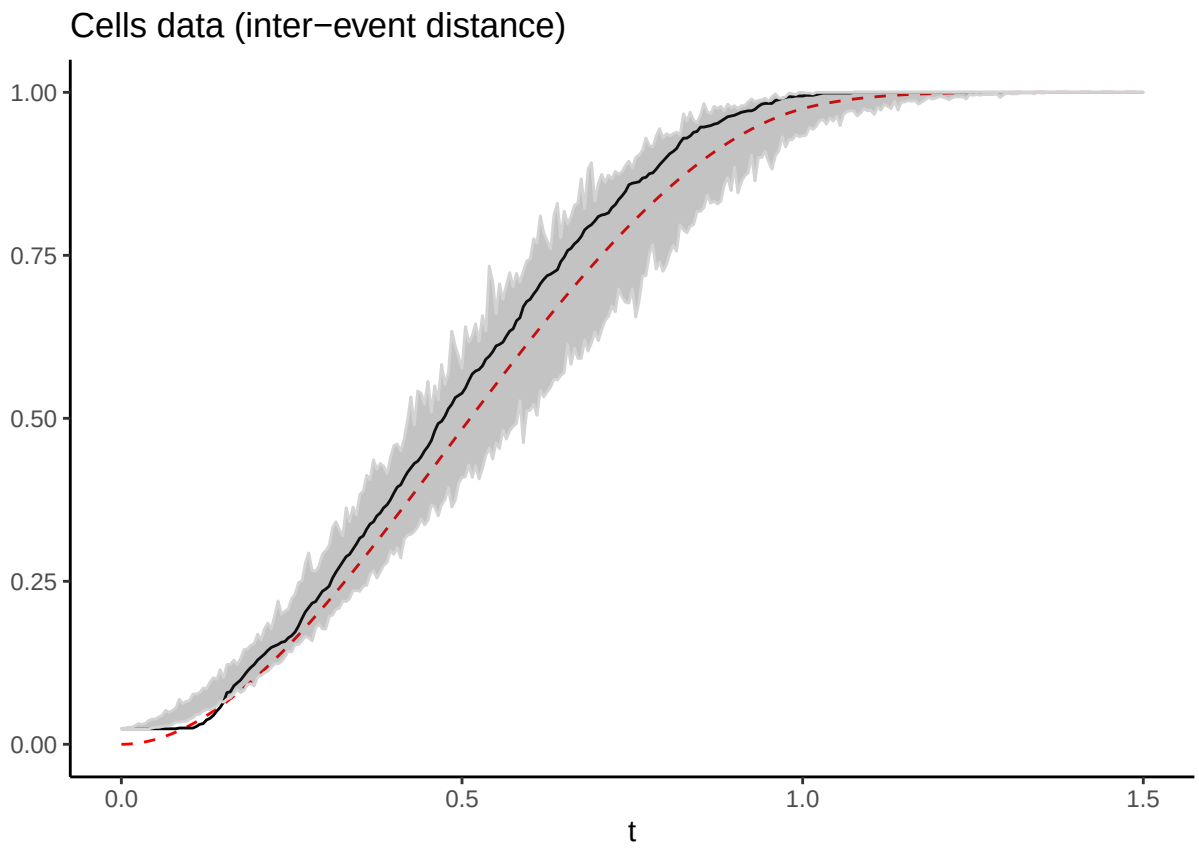
```
# area type: square
cells$window
```

```
## window: rectangle = [0, 1] x [0, 1] units
```

- $H(t)$ $vs.$ $\hat{H}(t)$

```r
res.c = data.frame(
  t = seq(0,1.5,0.005),
  U = NA,
  L = NA
)

res.c$H = CDF_H(res.c$t,'square')
res.c$H_hat = ECDF_H(cells,res.c$t)

# envelop
n = cells$n
A = owin(cells$window$xrange, cells$window$yrange)
s = 100
set.seed(42)

for (j in 1:nrow(res.c)) {
  t = res.c[j,'t']
  Hi_hat = rep(NA,s-1)

  for (i in 1:(s-1)) {
    dat_null = runifpoint(n,win = A)
    Hi_hat[i] = ECDF_H(dat_null,t)
  }

  res.c[j,'U'] = max(Hi_hat)
  res.c[j,'L'] = min(Hi_hat)
}
```

```r
res.c %>%
  ggplot() +
  geom_line(aes(x = t,y = H),col = 'red',linetype = 'dashed') +
  geom_line(aes(x = t,y = H_hat)) +
  geom_ribbon(aes(x = t,ymin = L,ymax = U),col = 'lightgray',alpha = 0.3) +
  theme_classic() +
  ggtitle("Cells data (inter-event distance)") +
  ylab("")
```

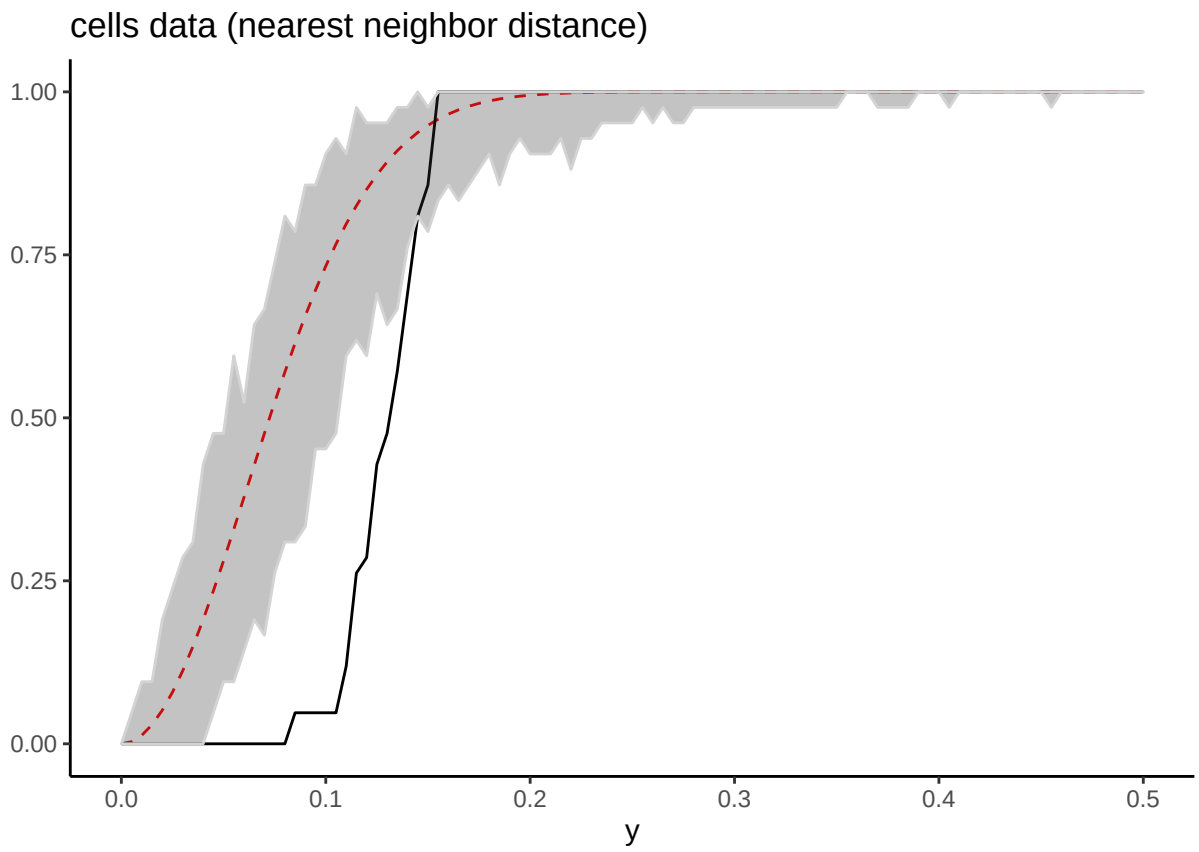


- $G(y)$ *vs.* $\hat{G}(y)$

```
res.c = data.frame(
  y = seq(0,0.5,0.005),
  U = NA,
  L = NA
)
res.c$G = CDF_G(cells,res.c$y)
res.c$G_hat = ECDF_G(cells,res.c$y)

# envelop
n = cells$n
A = owin(cells$window$xrange, cells$window$yrange)
s = 100
set.seed(42)

for (j in 1:nrow(res.c)) {
  y = res.c[j,'y']
  Gi_hat = rep(NA,s-1)

  for (i in 1:(s-1)) {
    dat_null = runifpoint(n,win = A)
    Gi_hat[i] = ECDF_G(dat_null,y)
  }

  res.c[j,'U'] = max(Gi_hat)
```

```
  res.c[j,'L'] = min(Gi_hat)
}
```

```
res.c %>%
  ggplot() +
  geom_line(aes(x = y,y = G),col = 'red',linetype = 'dashed') +
  geom_line(aes(x = y,y = G_hat)) +
  geom_ribbon(aes(x = y,ymin = L,ymax = U),col = 'lightgray',alpha = 0.3) +
  theme_classic() +
  ggtitle("cells data (nearest neighbor distance)") +
  ylab("")
```



두 가지 방법들에서 작은 t값에 대해 empirical CDF가 envelop를 벗어나 아래쪽에 위치해 있는 것을 확인할 수 있다. 이를 통해 cells data에 repulsive tendency가 있을 것이라 생각할 수 있다.

Problem 5.

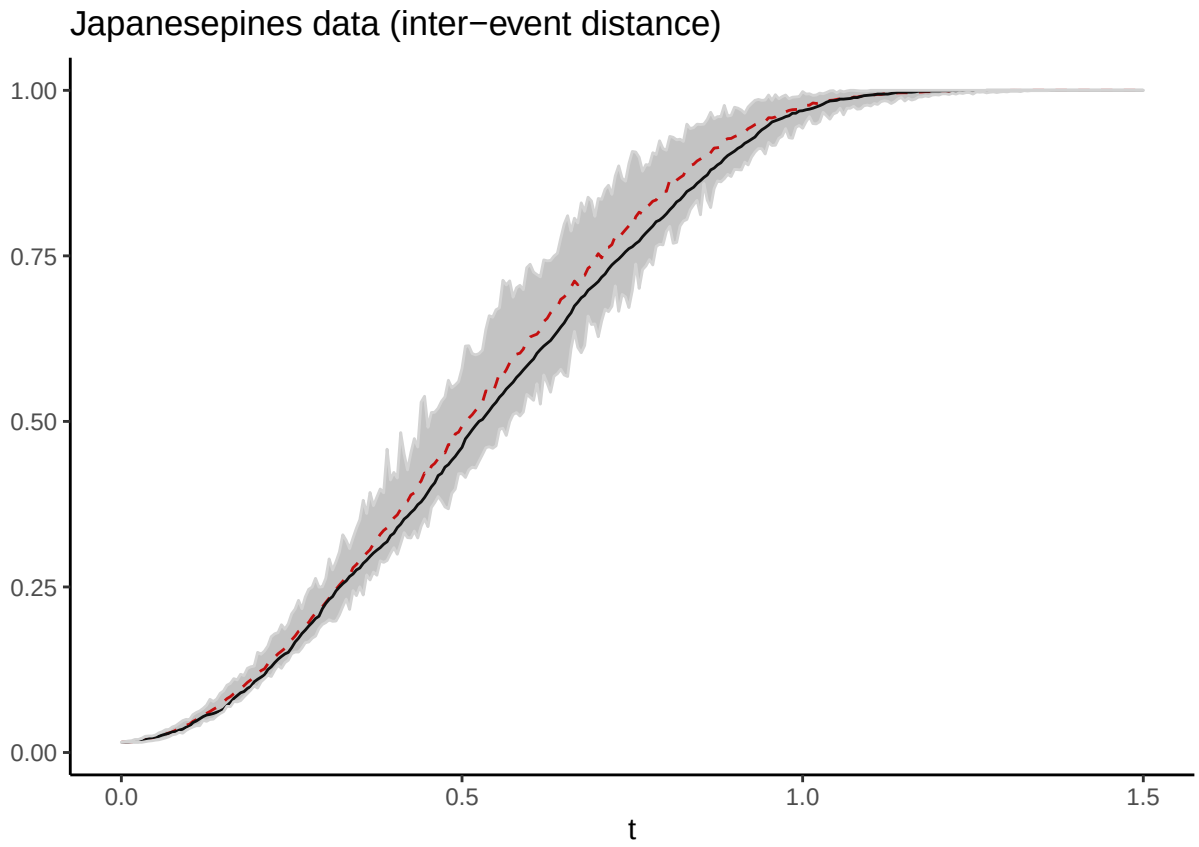
[20 pts] Repeat Q4 with ¯H (t) and ¯G(y) instead of H(t) and G(y).

[japanesepines data]

- $\bar{H}(t)$ *vs.* $\hat{H}(t)$

```
res.j = data.frame(
  t = seq(0,1.5,0.005),
  H_bar = NA,
  U = NA,
  L = NA
)

res.j$H_hat = ECDF_H(japanesepines,res.j$t)

# envelop
n = japanesepines$n
A = owin(japanesepines$window$xrange, japanesepines$window$yrange)
s = 100
set.seed(42)

for (j in 1:nrow(res.j)) {
  t = res.j[j,'t']
  Hi_hat = rep(NA,s-1)

  for (i in 1:(s-1)) {
    dat_null = runifpoint(n,win = A)
    Hi_hat[i] = ECDF_H(dat_null,t)
  }

  res.j[j,'H_bar'] = mean(Hi_hat)
  res.j[j,'U'] = max(Hi_hat)
  res.j[j,'L'] = min(Hi_hat)
}
```

```
res.j %>%
  ggplot() +
  geom_line(aes(x = t,y = H_bar),col = 'red',linetype = 'dashed') +
  geom_line(aes(x = t,y = H_hat)) +
  geom_ribbon(aes(x = t,ymin = L,ymax = U),col = 'lightgray',alpha = 0.3) +
  theme_classic() +
  ggtitle("Japanesepines data (inter-event distance)") +
  ylab("")
```

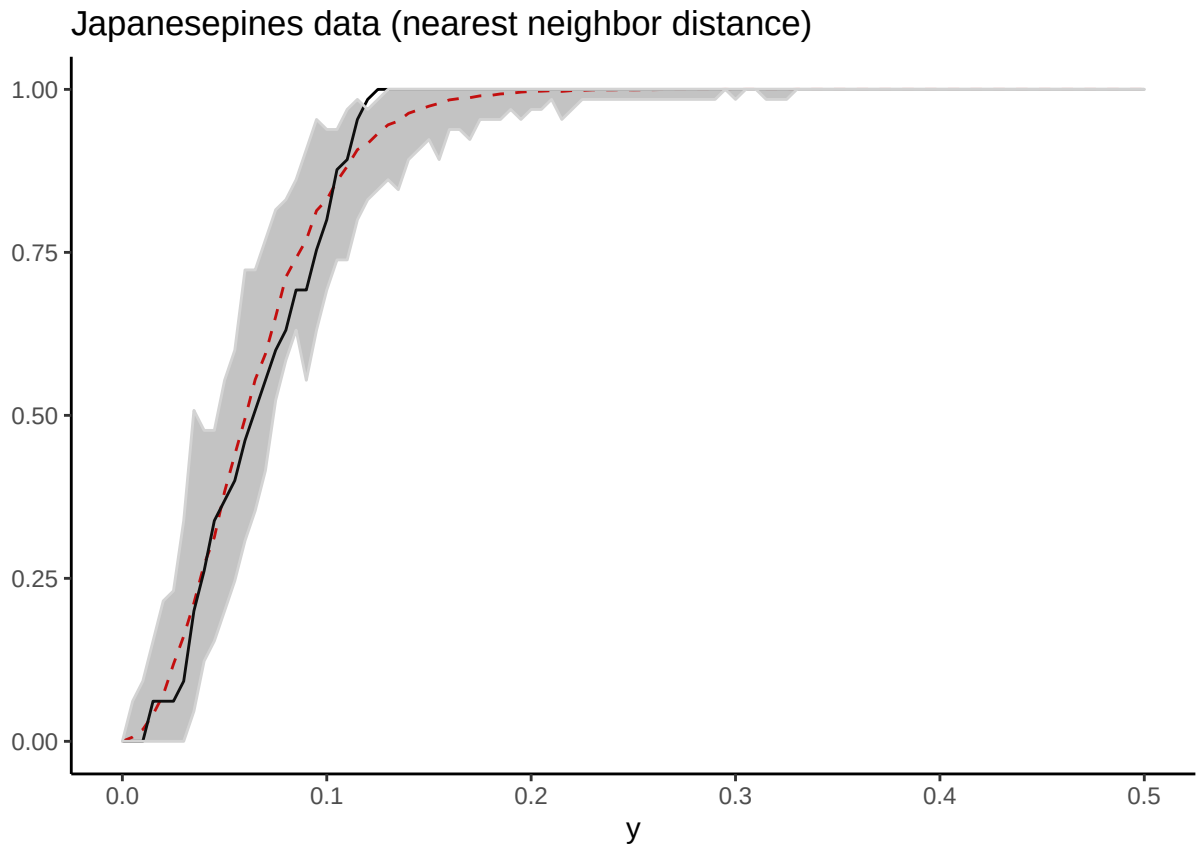


- $\bar{G}(y)$ *vs.* $\hat{G}(y)$

```
res.j = data.frame(
  y = seq(0,0.5,0.005),
  G_bar = NA,
  U = NA,
  L = NA
)

res.j$G_hat = ECDF_G(japanesepines,res.j$y)

# envelop
n = japanesepines$n
A = owin(japanesepines$window$xrange, japanesepines$window$yrange)
s = 100
set.seed(42)

for (j in 1:nrow(res.j)) {
  y = res.j[j,'y']
  Gi_hat = rep(NA,s-1)

  for (i in 1:(s-1)) {
    dat_null = runifpoint(n,win = A)
    Gi_hat[i] = ECDF_G(dat_null,y)
  }
```

```
  res.j[j,'G_bar'] = mean(Gi_hat)
  res.j[j,'U'] = max(Gi_hat)
  res.j[j,'L'] = min(Gi_hat)
}
```

```
res.j %>%
  ggplot() +
  geom_line(aes(x = y,y = G_bar),col = 'red',linetype = 'dashed') +
  geom_line(aes(x = y,y = G_hat)) +
  geom_ribbon(aes(x = y,ymin = L,ymax = U),col = 'lightgray',alpha = 0.3) +
  theme_classic() +
  ggtitle("Japanesepines data (nearest neighbor distance)") +
  ylab("")
```



3번에서와 마찬가지로, $\hat{H}(t)$ 가 $\bar{H}(t)$ 의 분포와 크게 차이나지 않았으며, 전체적으로 envelop를 벗어나지 않고 있었다. 따라서 CSR이 성립하지 않다고 볼 근거가 없다.

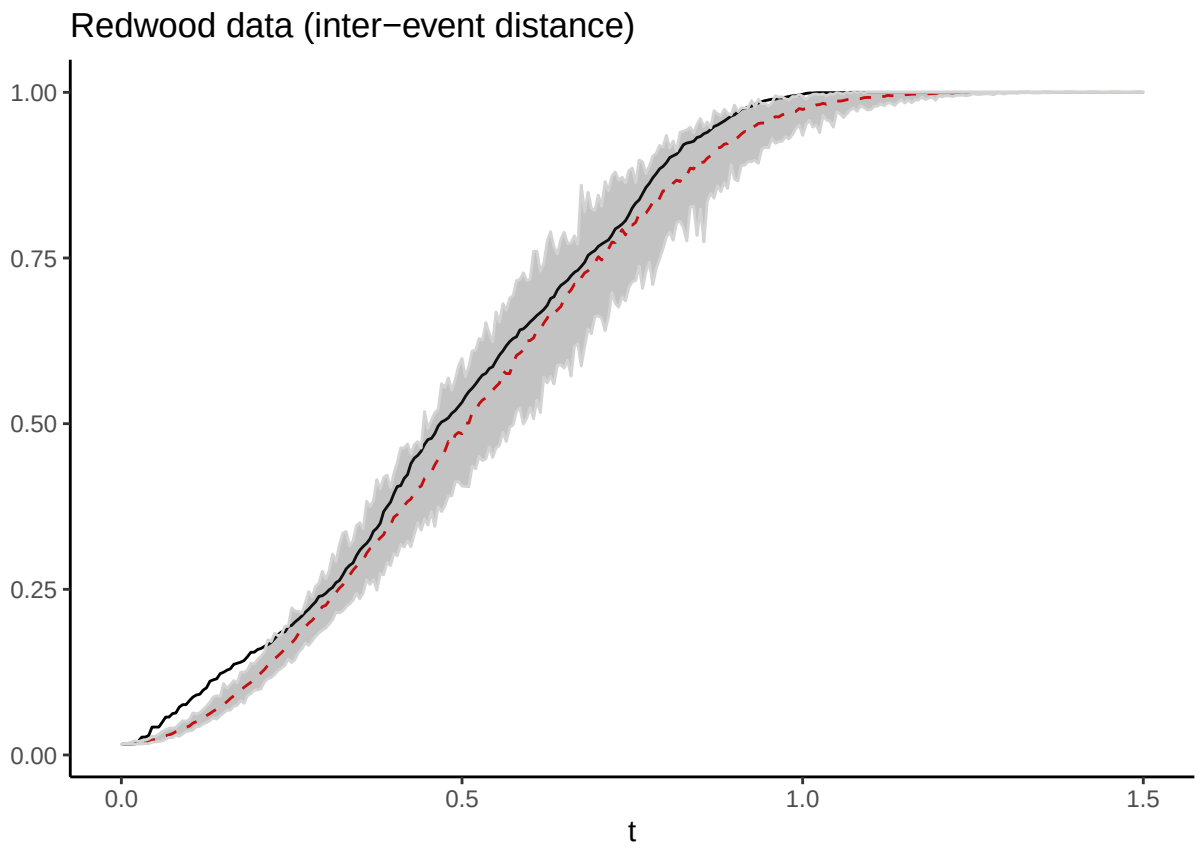
[redwood data]

- $\bar{H}(t)$ *vs.* $\hat{H}(t)$

```
res.r = data.frame(
  t = seq(0,1.5,0.005),
  H_bar = NA,
  U = NA,
```

```
  L = NA
)

res.r$H_hat = ECDF_H(redwood,res.r$t)

# envelop
n = redwood$n
A = owin(redwood$window$xrange, redwood$window$yrange)
s = 100
set.seed(42)

for (j in 1:nrow(res.r)) {
  t = res.r[j,'t']
  Hi_hat = rep(NA,s-1)

  for (i in 1:(s-1)) {
    dat_null = runifpoint(n,win = A)
    Hi_hat[i] = ECDF_H(dat_null,t)
  }

  res.r[j,'H_bar'] = mean(Hi_hat)
  res.r[j,'U'] = max(Hi_hat)
  res.r[j,'L'] = min(Hi_hat)
}
```

```
res.r %>%
  ggplot() +
  geom_line(aes(x = t,y = H_bar),col = 'red',linetype = 'dashed') +
  geom_line(aes(x = t,y = H_hat)) +
  geom_ribbon(aes(x = t,ymin = L,ymax = U),col = 'lightgray',alpha = 0.3) +
  theme_classic() +
  ggtitle("Redwood data (inter-event distance)") +
  ylab("")
```

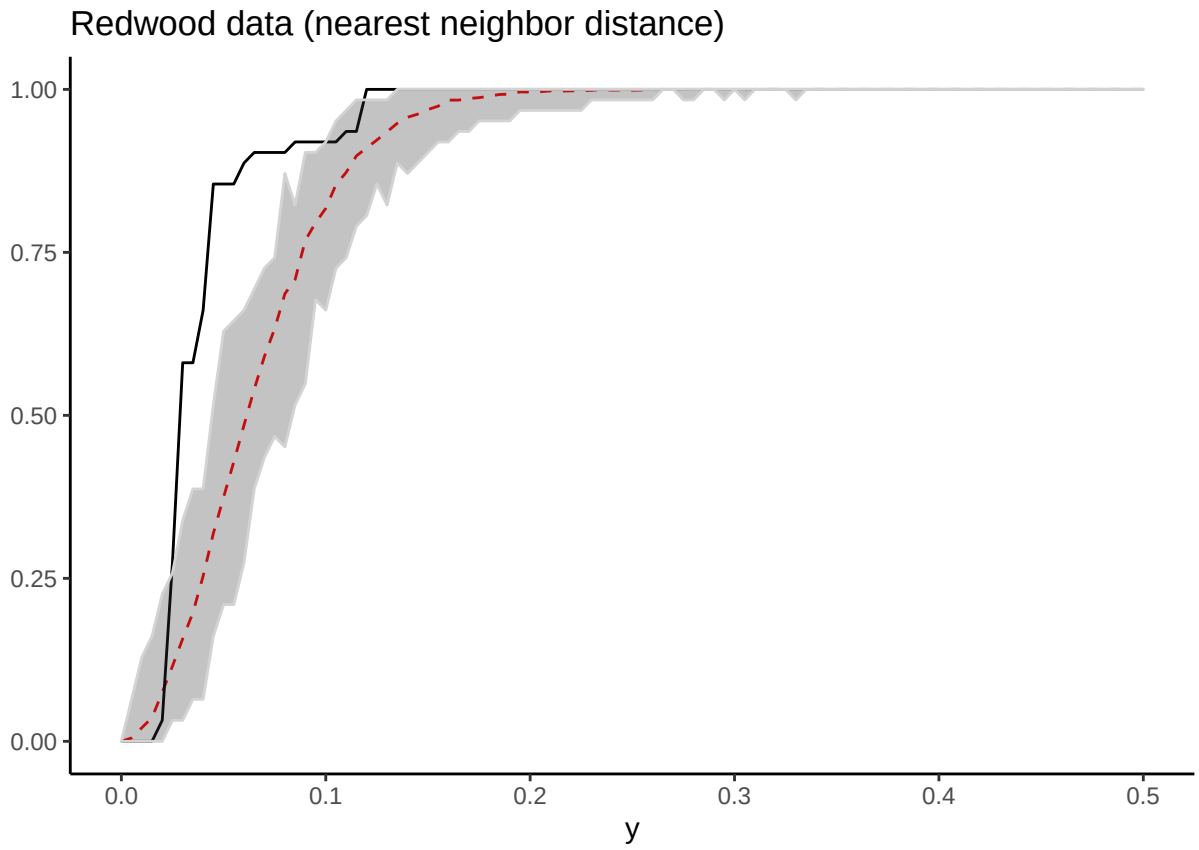


- $\bar{G}(y)$ *vs.* $\hat{G}(y)$

```
res.r = data.frame(
  y = seq(0,0.5,0.005),
  G_bar = NA,
  U = NA,
  L = NA
)

res.r$G_hat = ECDF_G(redwood,res.r$y)

# envelop
n = redwood$n
A = owin(redwood$window$xrange, redwood$window$yrange)
s = 100
set.seed(42)

for (j in 1:nrow(res.r)) {
  y = res.r[j,'y']
  Gi_hat = rep(NA,s-1)

  for (i in 1:(s-1)) {
    dat_null = runifpoint(n,win = A)
    Gi_hat[i] = ECDF_G(dat_null,y)
  }
```

```
  res.r[j,'G_bar'] = mean(Gi_hat)
  res.r[j,'U'] = max(Gi_hat)
  res.r[j,'L'] = min(Gi_hat)
}
```

```
res.r %>%
  ggplot() +
  geom_line(aes(x = y,y = G_bar),col = 'red',linetype = 'dashed') +
  geom_line(aes(x = y,y = G_hat)) +
  geom_ribbon(aes(x = y,ymin = L,ymax = U),col = 'lightgray',alpha = 0.3) +
  theme_classic() +
  ggtitle("Redwood data (nearest neighbor distance)") +
  ylab("")
```



3번에서와 마찬가지로, 작은 t값에 대해 empirical CDF가 envelop를 벗어나 위쪽에 위치해 있는 것을 확인할 수 있다. 이를 통해 redwood data에 aggregating tendency가 있을 것이라 생각할 수 있다.

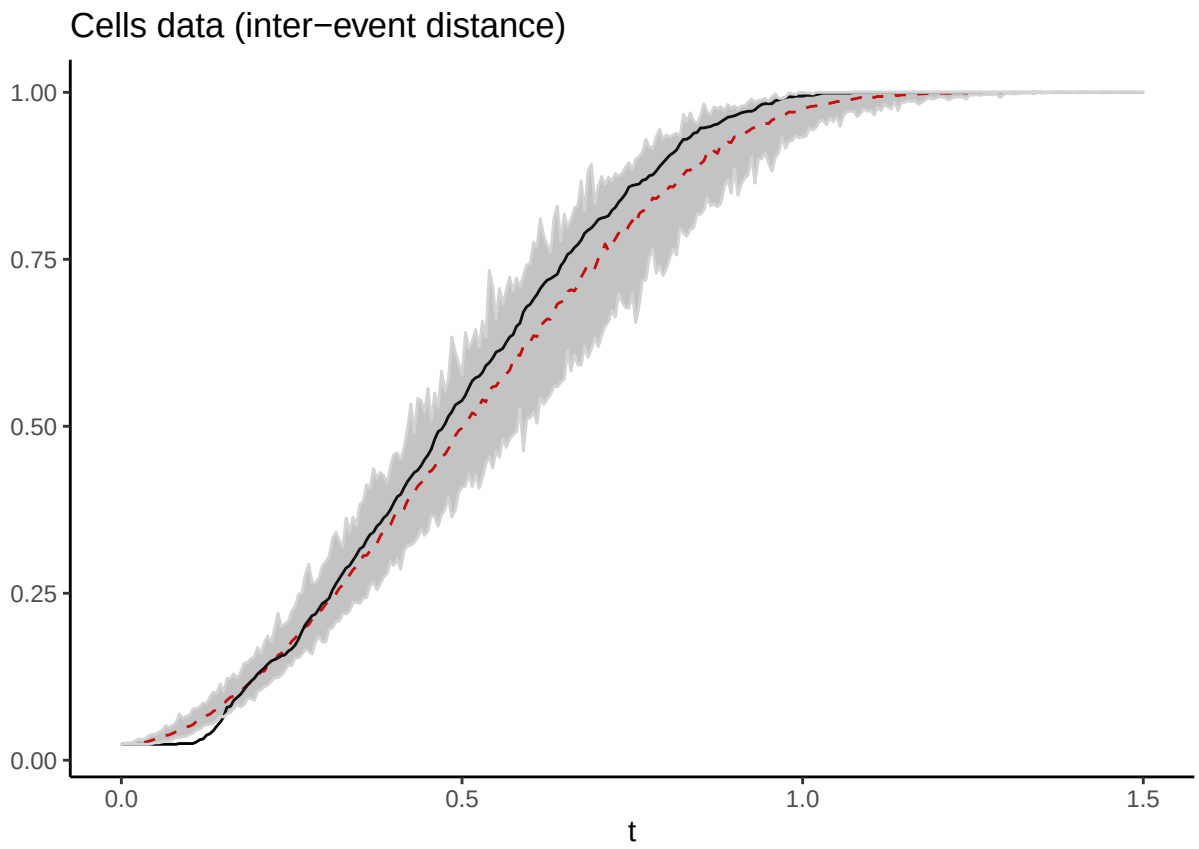
[cells data]

- $\bar{H}(t)$ *vs.* $\hat{H}(t)$

```
res.c = data.frame(
  t = seq(0,1.5,0.005),
  H_bar = NA,
  U = NA,
```

```
  L = NA
)

res.c$H_hat = ECDF_H(cells,res.c$t)

# envelop
n = cells$n
A = owin(cells$window$xrange, cells$window$yrange)
s = 100
set.seed(42)

for (j in 1:nrow(res.c)) {
  t = res.c[j,'t']
  Hi_hat = rep(NA,s-1)

  for (i in 1:(s-1)) {
    dat_null = runifpoint(n,win = A)
    Hi_hat[i] = ECDF_H(dat_null,t)
  }

  res.c[j,'H_bar'] = mean(Hi_hat)
  res.c[j,'U'] = max(Hi_hat)
  res.c[j,'L'] = min(Hi_hat)
}
```

```
res.c %>%
  ggplot() +
  geom_line(aes(x = t,y = H_bar),col = 'red',linetype = 'dashed') +
  geom_line(aes(x = t,y = H_hat)) +
  geom_ribbon(aes(x = t,ymin = L,ymax = U),col = 'lightgray',alpha = 0.3) +
  theme_classic() +
  ggtitle("Cells data (inter-event distance)") +
  ylab("")
```

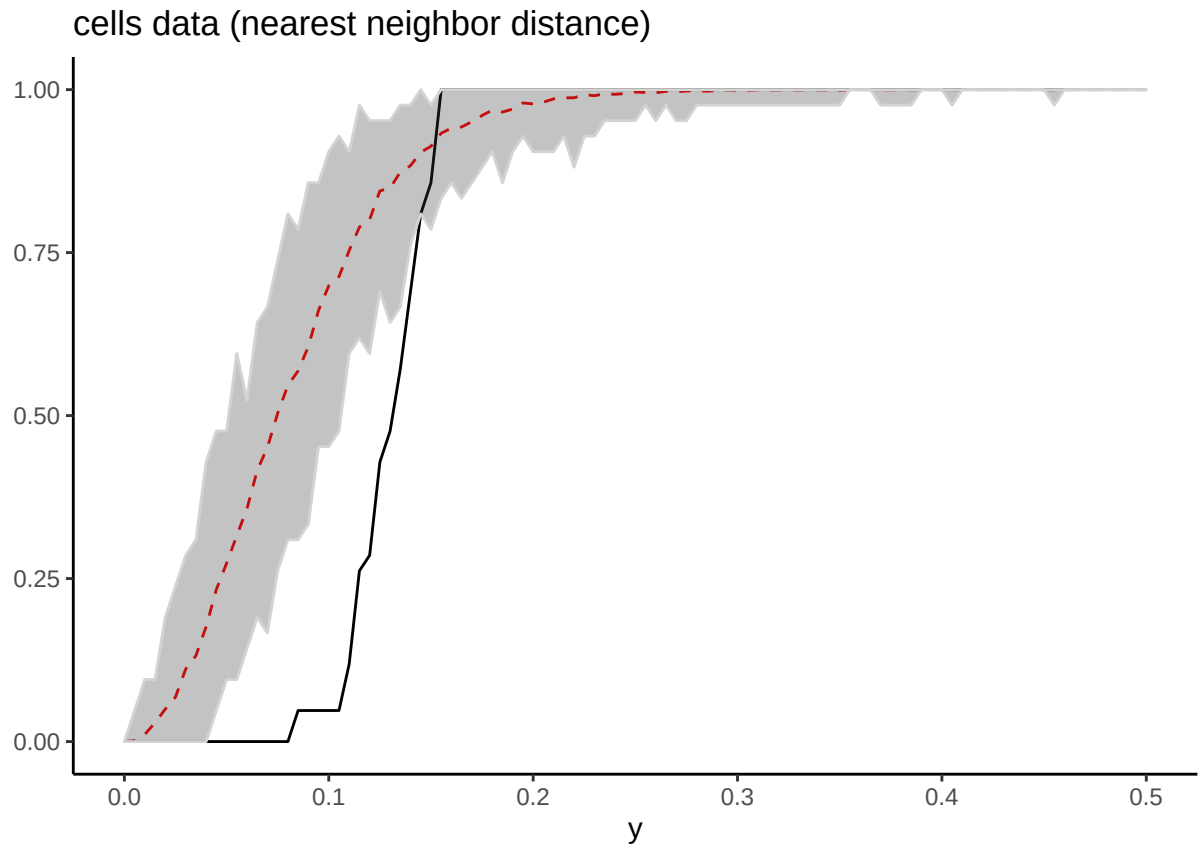


- $\bar{G}(y)$ *vs.* $\hat{G}(y)$

```
res.c = data.frame(
  y = seq(0,0.5,0.005),
  G_bar = NA,
  U = NA,
  L = NA
)

res.c$G_hat = ECDF_G(cells,res.c$y)

# envelop
n = cells$n
A = owin(cells$window$xrange, cells$window$yrange)
s = 100
set.seed(42)

for (j in 1:nrow(res.c)) {
  y = res.c[j,'y']
  Gi_hat = rep(NA,s-1)

  for (i in 1:(s-1)) {
    dat_null = runifpoint(n,win = A)
    Gi_hat[i] = ECDF_G(dat_null,y)
  }
```

```
  res.c[j,'G_bar'] = mean(Gi_hat)
  res.c[j,'U'] = max(Gi_hat)
  res.c[j,'L'] = min(Gi_hat)
}
```

```
res.c %>%
  ggplot() +
  geom_line(aes(x = y,y = G_bar),col = 'red',linetype = 'dashed') +
  geom_line(aes(x = y,y = G_hat)) +
  geom_ribbon(aes(x = y,ymin = L,ymax = U),col = 'lightgray',alpha = 0.3) +
  theme_classic() +
  ggtitle("cells data (nearest neighbor distance)") +
  ylab("")
```



3번에서와 마찬가지로, 작은 t값에 대해 empirical CDF가 envelop를 벗어나 아래쪽에 위치해 있는 것을 확인할 수 있다. 이를 통해 cells data에 repulsive tendency가 있을 것이라 생각할 수 있다.

Problem 6.

[20 pts] Do the MC test for CSR using u1 = ˆH (t0) for an appropriate t0 and the MC test for CSR using u1. Discuss your findings.

▶ MC test for CSR using $u_1 = \hat{H}(t_0)$ for appropriate t_0

```
U_from_H = function(data,t0,s) {
  n = data$n
  A = owin(data$window$xrange, data$window$yrange)
  U = rep(NA,s)
  set.seed(42)

  for (i in 1:s) {
    if (i == 1) {data_i = data}
    else {data_i = runifpoint(n,win = A)}
    U[i] = ECDF_H(data_i,t0)
  }
  return(U)
}
```

▶ MC test for CSR using $u_1 = \int (\hat{G}_1(y) - \bar{G}_1(y))^2 dy$

```
U_from_G = function(data,ymin,ymax,ylen,s) {
  n = data$n
  A = owin(data$window$xrange, data$window$yrange)
  y = seq(ymin,ymax,length.out = ylen)
  bw = (ymax-ymin)/ylen

  # row: event number / col: y
  G_hat = matrix(NA,nrow = s,ncol = ylen)
  G_bar = matrix(0,nrow = s,ncol = ylen)
  U = rep(NA,s)
  set.seed(42)

  for (i in 1:s) {
    if (i == 1) {data_i = data}
    else {data_i = runifpoint(n,win = A)}
    Di = pairdist(data_i)
    diag(Di) = max(Di)
    yi = apply(Di,1,min)

    for (m in 1:ylen) {
      G_hat[i,m] = mean(yi <= y[m])

      # G_bar[i,m] = 1/(s-1) * {G_hat[1,m] + G_hat[2,m] + ... + G_hat[s,m] - G_hat[i,m]}
      G_bar[i,m] = G_bar[i,m] - G_hat[i,m]
    }
  }

  for (m in 1:ylen) {G_bar[,m] = G_bar[,m] + sum(G_hat[,m])}
  G_bar = G_bar / (s-1)
```

```
  for (i in 1:s) {U[i] = sum((G_hat[i,]-G_bar[i,])^2)*bw}
  return(U)
}
```

$(size\ of\ test) = 0.05$ 로 두고 CSR test를 진행한다.

[japanesepines data]

1. using $u_1 = \hat{H}(t_0)$ for appropriate t_0
: Problem 4 의 결과를 참고하여 $t_0 = 0.5$ 로 두었다.

```
U = U_from_H(japanesepines,0.5,100)

U1 = U[1]
U.sorted = sort(U,decreasing = T)
cat("U1 = ",U1,"\tU(5) = ",U.sorted[5],"\n",
    "=> U1 >= U(5) : ",U1 >= U.sorted[5])
```

```
## U1 =  0.4608284  U(5) =  0.5540828
##  => U1 >= U(5) :  FALSE
```

2. using $u_1 = \int (\hat{G}_1(y) - \bar{G}_1(y))^2 dy$

```
U = U_from_G(japanesepines,0,sqrt(2),10000,100)

U1 = U[1]
U.sorted = sort(U,decreasing = T)
cat("U1 = ",U1,"\tU(5) = ",U.sorted[5],"\n",
    "=> U1 >= U(5) : ",U1 >= U.sorted[5])
```

```
## U1 =  0.0003214859   U(5) =  0.0008378253
##  => U1 >= U(5) :  FALSE
```

두가지 검정 모두 귀무가설을 기각하지 못한다. 따라서 유의수준 0.05에서 CSR이 성립하지 않는다고 볼 충분한 근거가 없다.

[redwood data]

1. using $u_1 = \hat{H}(t_0)$ for appropriate t_0
: Problem 4 의 결과를 참고하여 $t_0 = 0.1,\ 0.5$ 로 두었다.

```
# t0 = 0.1
U = U_from_H(redwood,0.1,100)

U1 = U[1]
U.sorted = sort(U,decreasing = T)
cat("U1 = ",U1,"\tU(5) = ",U.sorted[5],"\n",
    "=> U1 >= U(5) : ",U1 >= U.sorted[5])
```

```
## U1 =  0.08220604     U(5) =  0.05150884
##  => U1 >= U(5) :  TRUE
```

```
# t0 = 0.5
U = U_from_H(redwood,0.5,100)

U1 = U[1]
U.sorted = sort(U,decreasing = T)
cat("U1 = ",U1,"\tU(5) = ",U.sorted[5],"\n",
    "=> U1 >= U(5) : ",U1 >= U.sorted[5])
```

```
## U1 =  0.5322581  U(5) =  0.5473465
##  => U1 >= U(5) :  FALSE
```

$t_0 = 0.1$일 때는 귀무가설을 기각하였지만, $t_0 = 0.5$일 때는 귀무가설을 기각하지 못했다. Problem 4의 그래프에서 작은 t값에 대해 empirical CDF가 envelop를 벗어나 있었는데, 이와 일맥상통한다. 다음으로 전체적인 t에 대해 MC test를 하고자 한다.

2. using $u_1 = \int (\hat{G}_1(y) - \bar{G}_1(y))^2 dy$

```
U = U_from_G(redwood,0,sqrt(2),10000,100)

U1 = U[1]
U.sorted = sort(U,decreasing = T)
cat("U1 = ",U1,"\tU(5) = ",U.sorted[5],"\n",
    "=> U1 >= U(5) : ",U1 >= U.sorted[5])
```

```
## U1 =  0.009034281     U(5) =  0.001271022
##  => U1 >= U(5) :  TRUE
```

귀무가설을 기각하였다. 따라서 유의수준 0.05에서 CSR이 성립하지 않는다고 볼 수 있다.

[cells data]

1. using $u_1 = \hat{H}(t_0)$ for appropriate t_0
: Problem 4 의 결과를 참고하여 $t_0 = 0.1,\ 0.5$ 로 두었다.

```
# t0 = 0.1
U = U_from_H(cells,0.1,100)

U1 = U[1]
U.sorted = sort(U,decreasing = T)
cat("U1 = ",U1,"\tU(5) = ",U.sorted[5],"\n",
    "=> U1 >= U(5) : ",U1 >= U.sorted[5])
```

```
## U1 =  0.02494331     U(5) =  0.06575964
##  => U1 >= U(5) :  FALSE
```

```r
# t0 = 0.5
U = U_from_H(cells,0.5,100)

U1 = U[1]
U.sorted = sort(U,decreasing = T)
cat("U1 = ",U1,"\tU(5) = ",U.sorted[5],"\n",
    "=> U1 >= U(5) : ",U1 >= U.sorted[5])
```

```
## U1 =  0.5385488  U(5) =  0.5668934
##  => U1 >= U(5) :  FALSE
```

$t_0 = 0.1,\ 0.5$일 때 모두 귀무가설을 기각하지 못했다. 다음으로 전체적인 t에 대해 MC test를 하고자 한다.

2. using $u_1 = \int (\hat{G}_1(y) - \bar{G}_1(y))^2 dy$

```r
U = U_from_G(cells,0,sqrt(2),10000,100)

U1 = U[1]
U.sorted = sort(U,decreasing = T)
cat("U1 = ",U1,"\tU(5) = ",U.sorted[5],"\n",
    "=> U1 >= U(5) : ",U1 >= U.sorted[5])
```

```
## U1 =  0.02384874     U(5) =  0.002368939
##  => U1 >= U(5) :  TRUE
```

귀무가설을 기각하였다. 따라서 위 검정에 의하면 유의수준 0.05에서 CSR이 성립하지 않는다고 볼 수 있다. Problem 4에서 inter-event distance에 기반한 그래프보다 nearest neighbor distance에 기반한 그래프에서 empirical CDF가 CSR 가정으로부터 많이 벗어나 있었기 때문에 이러한 결과가 발생한 것 같다.

Problem 7.

[20 pts] Without assuming homogeneity for each of three datasets japanesepines, reedwood, cells, estimate and plot K and L functions and compare them with Poisson point processes.

각 데이터들에 대해 inhomogeneity 를 가정하고, K-function 을 계산해주는 kinhom, L-function 을 계산해주는 Linhom 함수를 사용한다.

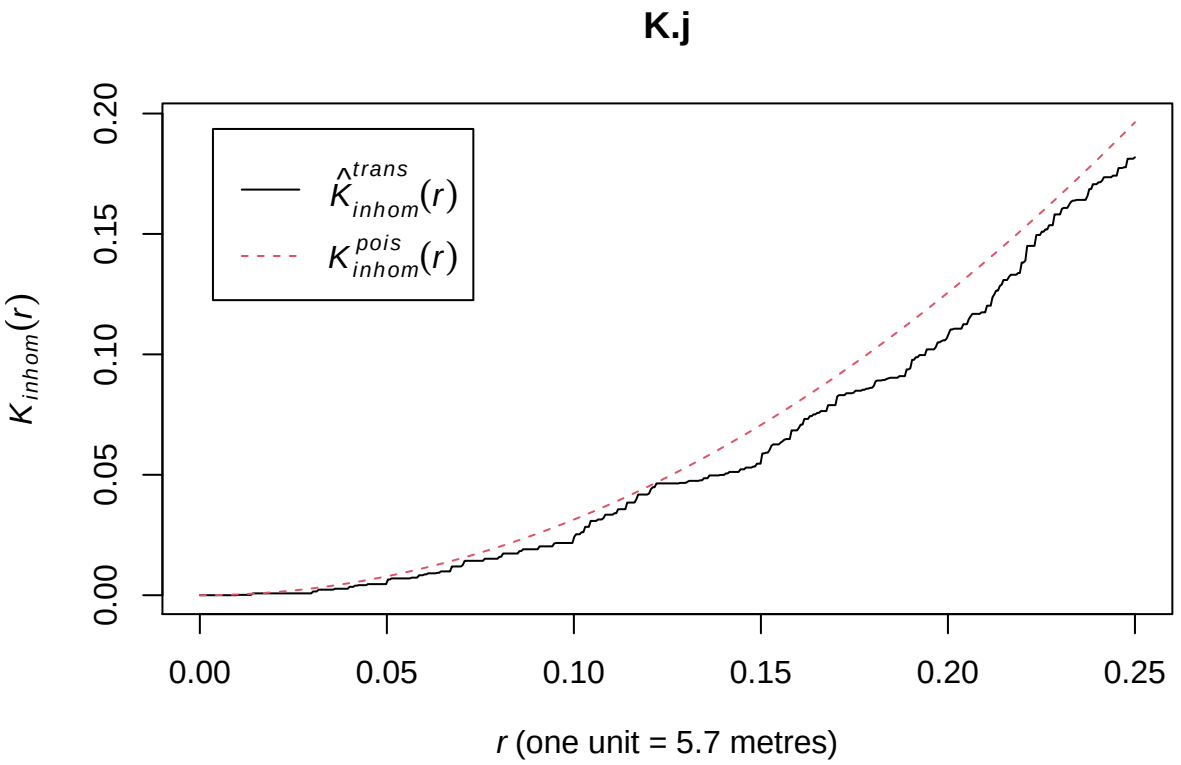
[japanesepines data]

▶ K function

```
K.j = Kinhom(japanesepines, correction = 'translate')
print(K.j$trans[1:30]) # estimate of K ftn
```

```
##  [1] 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000
##  [6] 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000
## [11] 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000
## [16] 0.0000000000 0.0000000000 0.0000000000 0.0000000000 0.0000000000
## [21] 0.0000000000 0.0001420933 0.0001420933 0.0001420933 0.0001420933
## [26] 0.0001420933 0.0001420933 0.0001420933 0.0001420933 0.0007637147
```

```
plot(K.j)
```

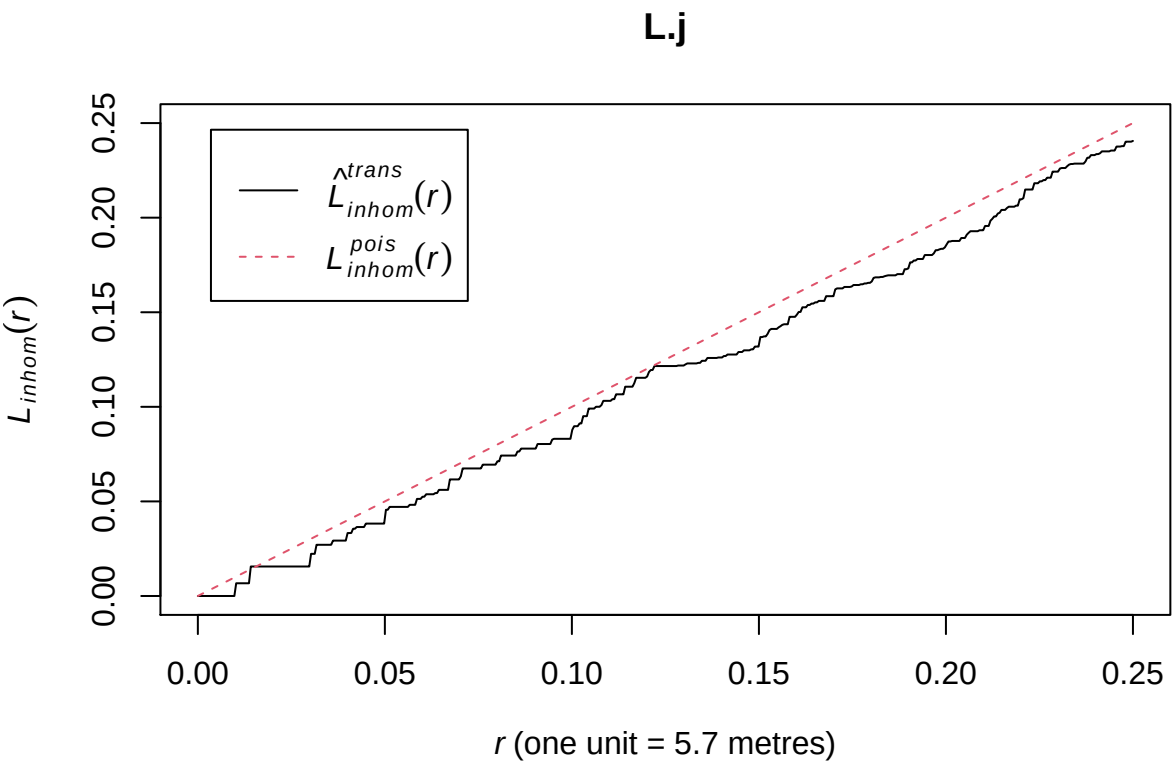


▶ L function

```
L.j = Linhom(japanesepines, correction = 'translate')
print(L.j$trans[1:30]) # estimate of L ftn
```

```
##  [1] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
##  [7] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
## [13] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
## [19] 0.000000000 0.000000000 0.000000000 0.006725304 0.006725304 0.006725304
## [25] 0.006725304 0.006725304 0.006725304 0.006725304 0.006725304 0.015591598
```

```
plot(L.j)
```



주어진 japanesepines 데이터로부터의 K-function, L-function 이 Poisson point process 하에서의 값들과 큰 차이를 보이지 않는다.

[redwood data]

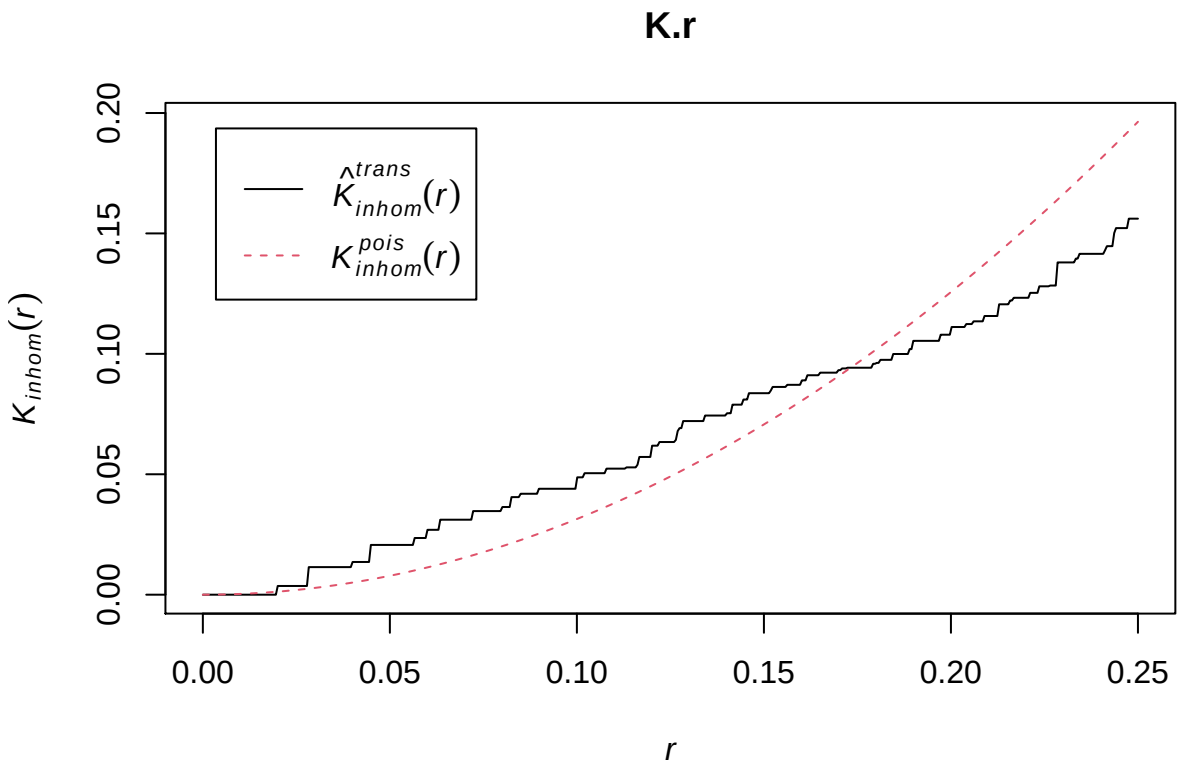
▶ K function

```
K.r = Kinhom(redwood, correction = 'translate')
print(K.r$trans[1:50]) # estimate of K ftn
```

```
##  [1] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
##  [7] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
## [13] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
```

```
## [19] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
## [25] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
## [31] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000
## [37] 0.000000000 0.000000000 0.000000000 0.000000000 0.000000000 0.003619559
## [43] 0.003619559 0.003619559 0.003619559 0.003619559 0.003619559 0.003619559
## [49] 0.003619559 0.003619559
```

```
plot(K.r)
```

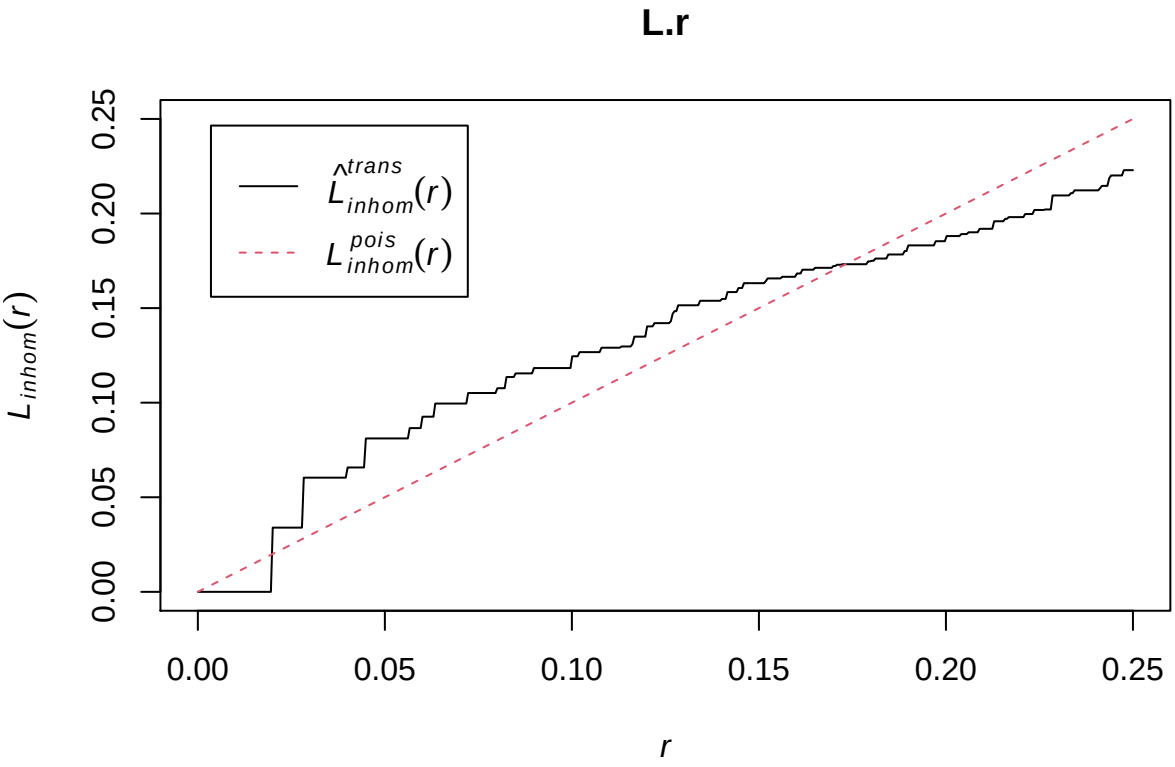


▶ L function

```
L.r = Linhom(redwood, correction = 'translate')
print(L.r$trans[1:50]) # estimate of L ftn
```

```
##  [1] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
##  [7] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## [13] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## [19] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## [25] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## [31] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
## [37] 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000 0.03394321
## [43] 0.03394321 0.03394321 0.03394321 0.03394321 0.03394321 0.03394321
## [49] 0.03394321 0.03394321
```

```
plot(L.r)
```



주어진 redwood 데이터로부터의 K-function, L-function 이 Poisson point process 하에서의 값들보다 전반적으로 더 크다는 것을 확인할 수 있다. 따라서 redwood 데이터에 aggregation tendency 가 있을 것이라 짐작해볼 수 있다.

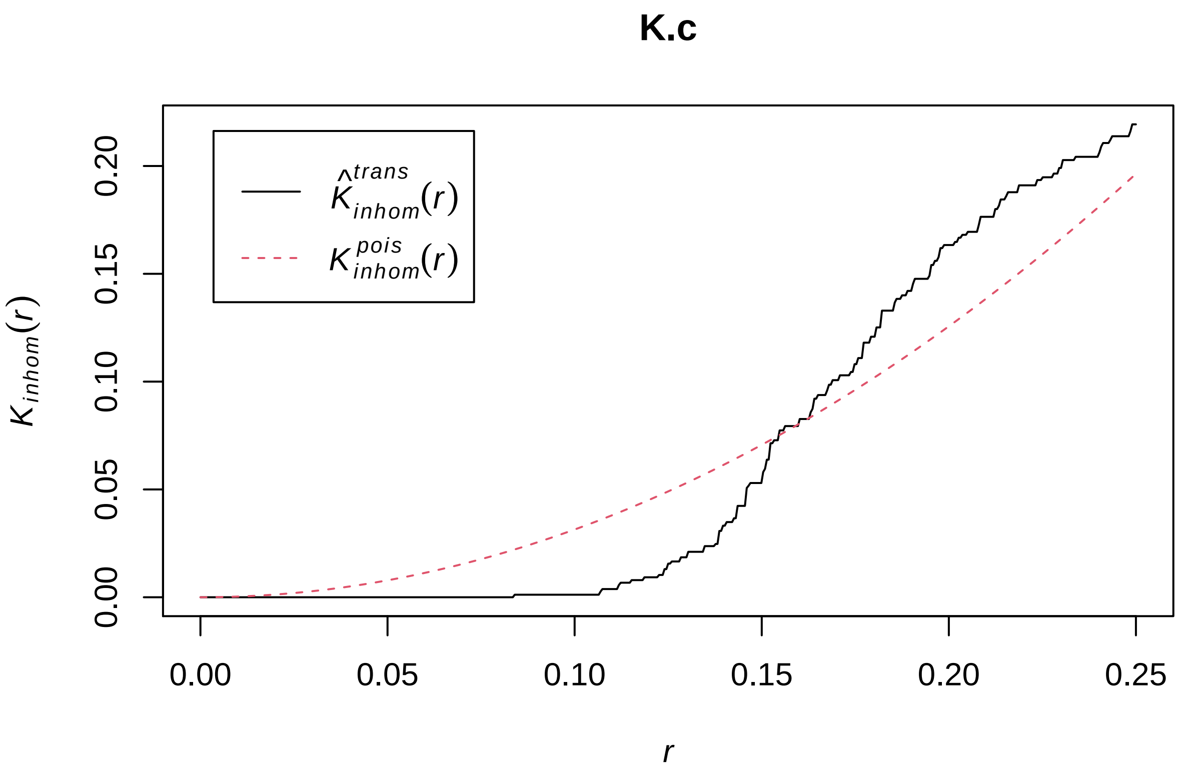
[cells data]

▶ K function

```
K.c = Kinhom(cells, correction = 'translate')
print(K.c$trans[1:50]) # estimate of K ftn
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [39] 0 0 0 0 0 0 0 0 0 0 0 0
```

```
plot(K.c)
```

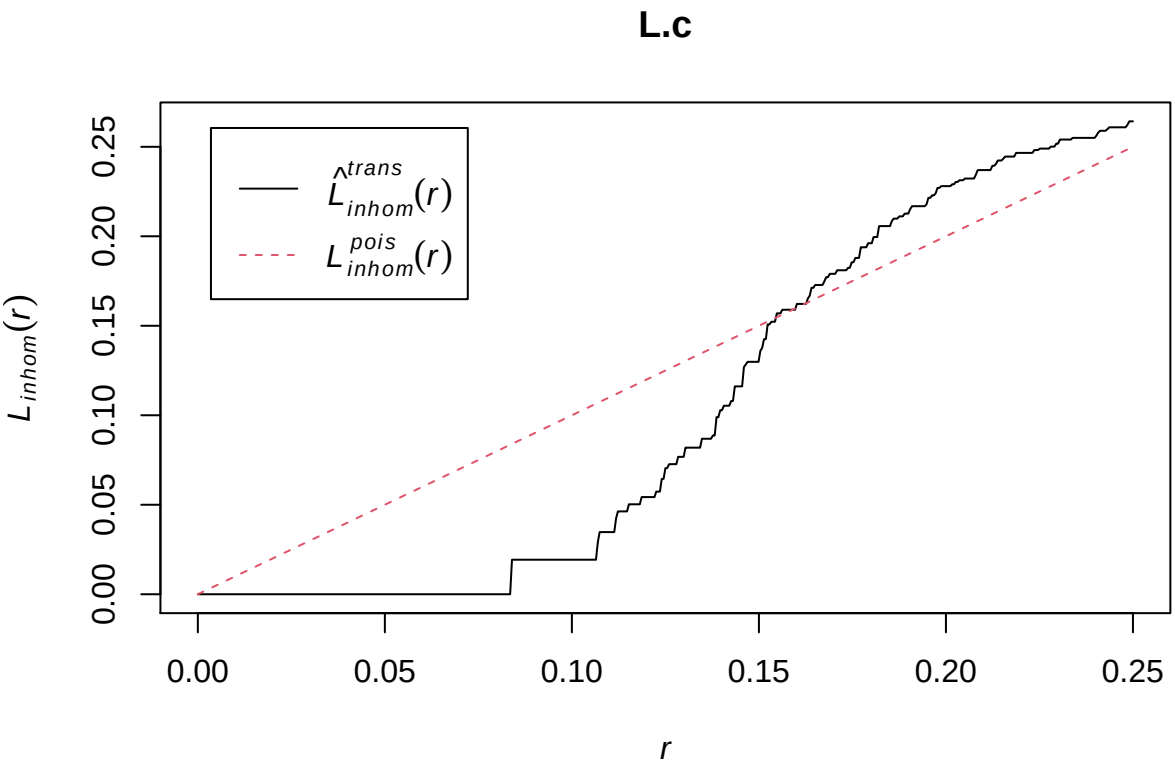


▶ L function

```
L.c = Linhom(cells, correction = 'translate')
print(L.c$trans[1:50]) # estimate of L ftn
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [39] 0 0 0 0 0 0 0 0 0 0 0 0
```

```
plot(L.c)
```



주어진 cells 데이터로부터의 K-function 이 Poisson point process 하에서의 값과 다른 패턴을 보이고 있으며, 특히 데이터로부터의 L-function 이 Poisson point process 하에서의 값보다 전반적으로 더 작다는 것을 확인할 수 있다. 따라서 cells 데이터에 repulsive tendency 가 있을 것이라 짐작해볼 수 있다.

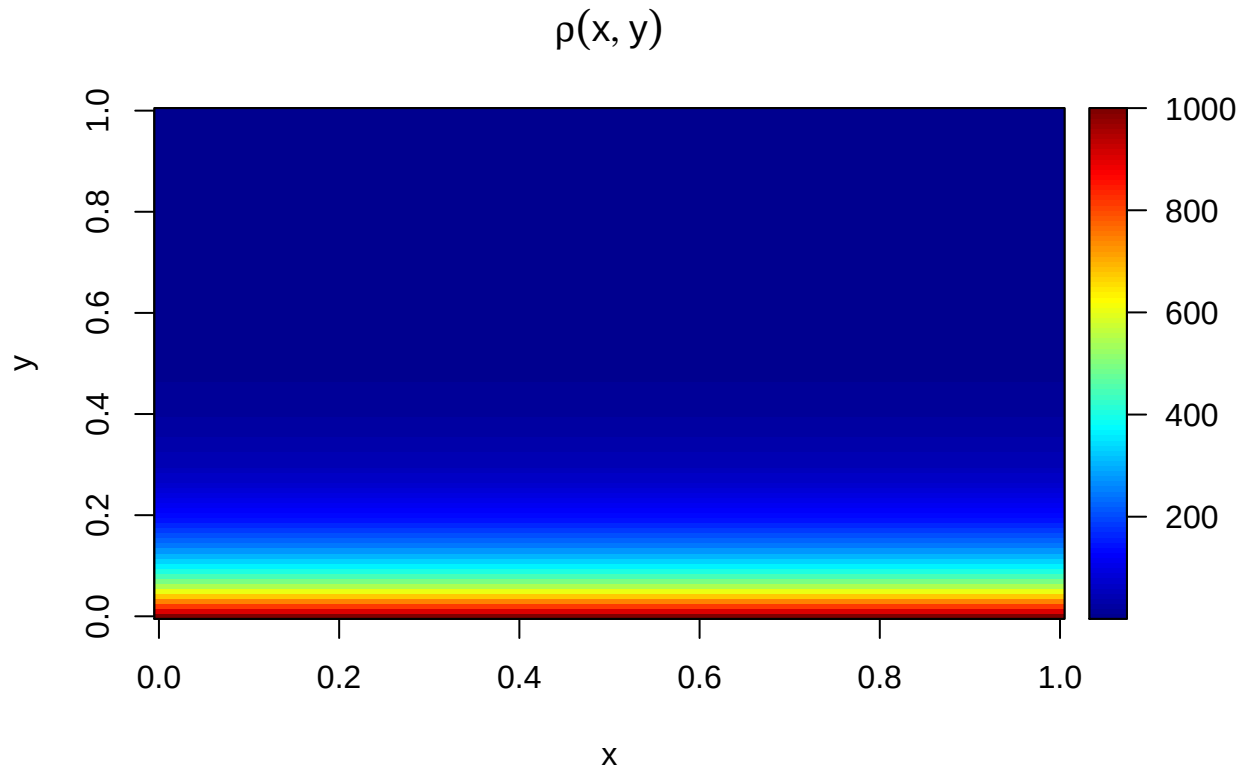
Problem 8.

[20 pts] Generate an inhomogeneous Poisson point process on [0, 1] × [0, 1] whose number of expected points is 100 and ρ(ξ1, ξ2) ∝ exp(−10ξ2). Then, using the generated data, calculate ˆρ(ξ) with an appropriate bandwidth b and compare with the true intensity function. ˆρ(ξ) (bias corrected nonparametric estiamte) is given in p. 55 of the lecture note 2. For this problem, use an Epanechnikov kernel, e(u) = (3/4)(1 − |u|2)1(|u| ≤ 1).

▶ True intensity function

Note that $\int_{[0,1]}\int_{[0,1]} ce^{-10y}dxdy = 100 \quad => \quad c = 1000$

```
# true intensity ftn
rho = function(x,y) {
  return(1000*exp(-10*y))
}

m = outer(seq(0,1,0.01),seq(0,1,0.01),
          Vectorize(rho))

image2D(m,main = expression(rho(x,y)))
```



▶ Estimated intensity function

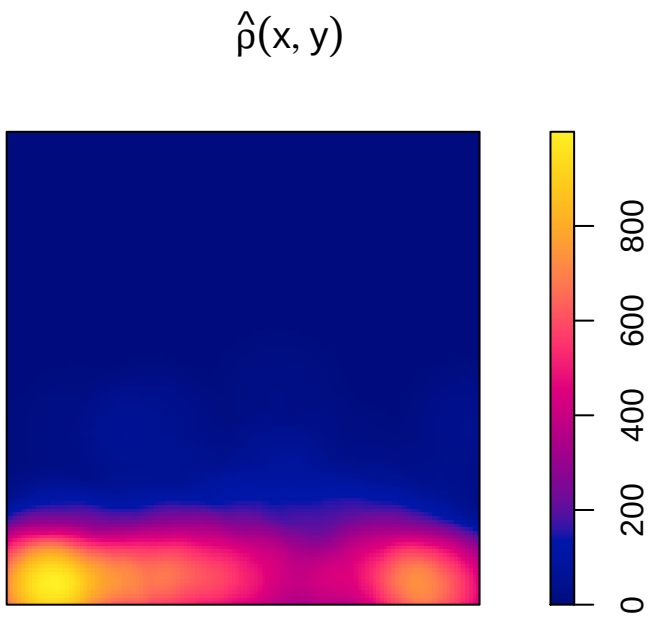
```r
# generated inhomogeneous ppp
set.seed(42)
ippp = rpoispp(lambda = rho,
               win = owin(c(0,1),c(0,1))) # inhomogeneous ppp

# appropriate bandwidth by cv
bw = LSCV.density(ippp)
```

```
## Searching for optimal h in [0.00319919209741694, 0.166666666666667]...Done.
```

```r
bw
```

```
## [1] 0.06683024
```

```r
rho.hat = density(ippp, sigma = bw, diggle = TRUE, kernel = 'epanechnikov')
plot(rho.hat, main = expression(hat(rho)(x,y)))
```



Epanechnikov kernel 을 이용하여 구한 $\hat{\rho}$ 은 ρ 와 비슷하게 작은 y값에 point 들이 모여있는 분포를 보였다. ρ 에서는 x축 방향으로 point 들이 균일하게 분포해 있었지만, $\hat{\rho}$ 은 x축 중앙보다 양극단에 점들이 좀 더 모여있는 분포를 보였다.