Analysis of Gordon Square data for hybrids paper

Adrian Baddeley

November 2, 2013

This `Sweave` script is part of the online supplementary material for the paper

Hybrids of Gibbs point processes and their implementation
Baddeley, Turner, Mateu and Bevan

This document was generated from the Sweave file `gordon.Rnw`.

1 Setup

Fix the random seed, to ensure repeatability.

```
> set.seed(42)
```

Number of iterations of each Metropolis-Hastings simulation run.

```
> Nrmh <- 5e5
```

Set this flag to TRUE if you make any changes to the code.

```
> recompute <- FALSE
```

Fix the spacing of dummy points, and the spatial resolution for images etc

```
> eps   <- 0.5  # about 0.5 x (50/0.5) x (70/0.5) = 7000 dummy points
> NGRID <- 128
```

Create directory for plot files

```
> if(!file.exists("pix-auto")) dir.create("pix-auto")
```

Load the required package including data.

```
> library(spatstat)
> if(versionstring.spatstat() < "1.34-1") {
+   stop("spatstat version 1.34-1 or later is required")
+ }
```

```
> sessionLibs()
```

```
Libraries loaded:
         spatstat 1.34-1
         polyclip 1.1-0
         tensor 1.5
         abind 1.4-0
         deldir 0.0-22
         mgcv 1.7-26
         nlme 3.1-111
```

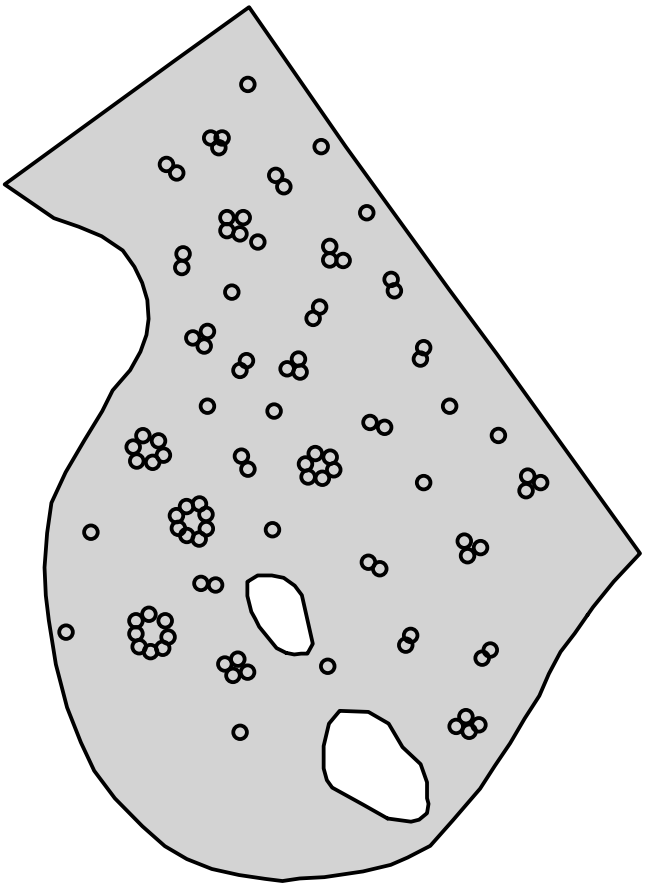
Fix plot parameters for different types of object. Main title is suppressed.

```
> Plot <- function(x, ...) {
+   specialpar <- list()
+   if(is.fv(x)) {
+     opa <- par(mar=rep(4,4)+0.1)
+     specialpar <- list(lwd=2, legend=FALSE)
+   } else if(is.ppp(x)) {
+     opa <- par(mar=rep(2,4))
+   } else if (is.owin(x) || is.im(x)) {
+     opa <- par(mar=rep(2,4))
+   } else {
+     opa <- par(mar=rep(4,4)+0.1)
+   }
+   do.call(plot, resolve.defaults(list(x=x), list(...),
+                                  list(main=""),
+                                  specialpar))
+   par(opa)
+ }
```

2 Data

The Gordon Square data

```
> PlotGordon <- function(X) {
+   Plot(as.owin(gordon), col="lightgrey", border="black", lwd=2)
+   plot(X, add=TRUE, lwd=2)
+ }

> PlotGordon(gordon)
```



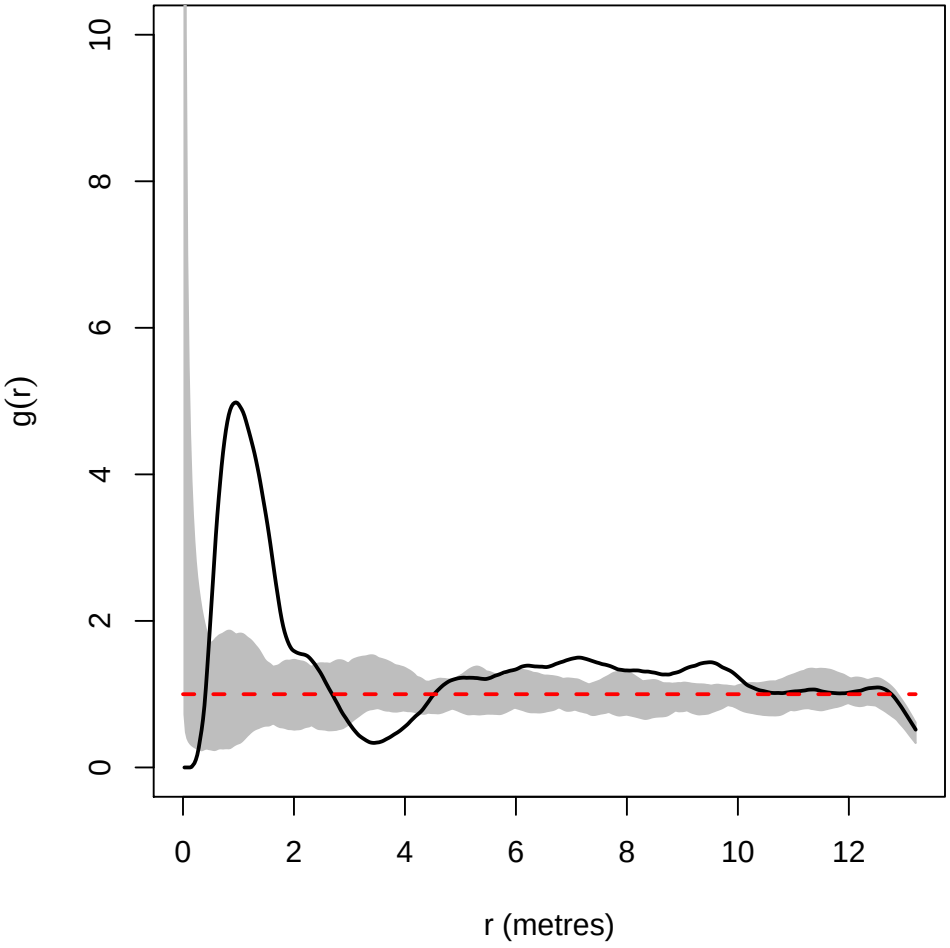
3 Initial examination

Compute pair correlation function etc

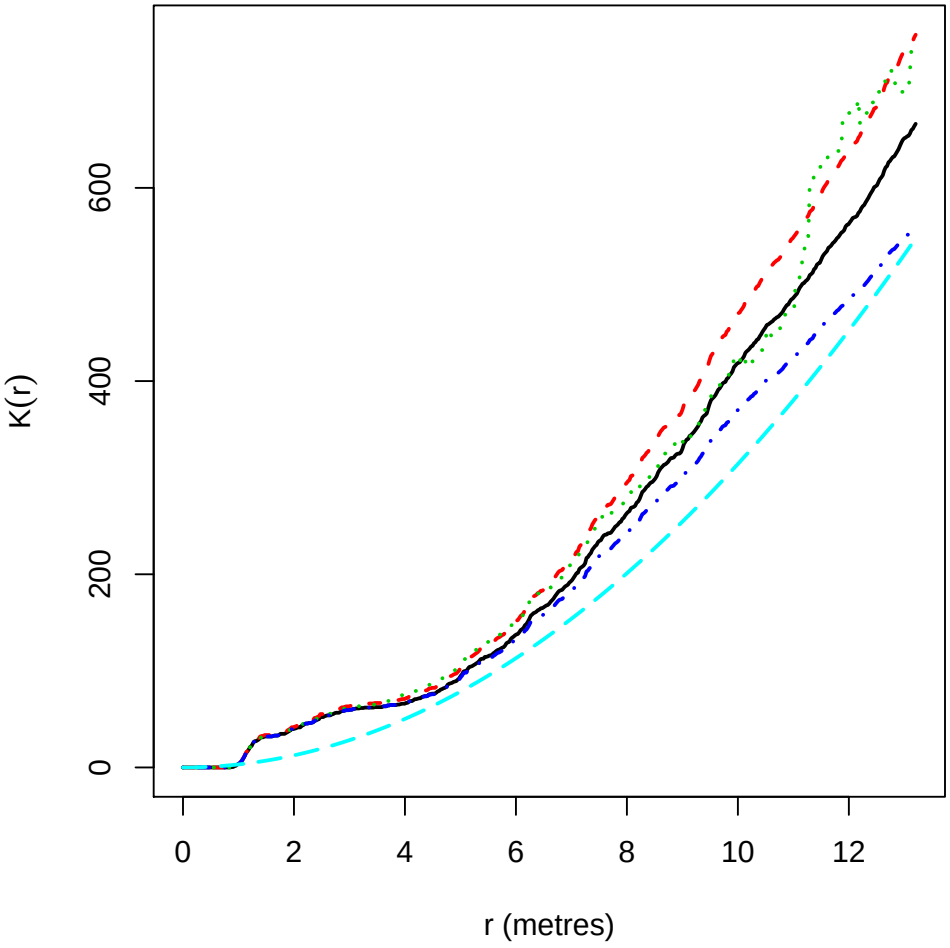
```
> pcfA <- envelope(gordon, pcf, nsim=39)
> KA <- Kest(gordon, correction=c("iso", "trans", "border", "none"))
> GA <- Gest(gordon, correction=c("km", "han", "rs", "none"))
```

Plot them.

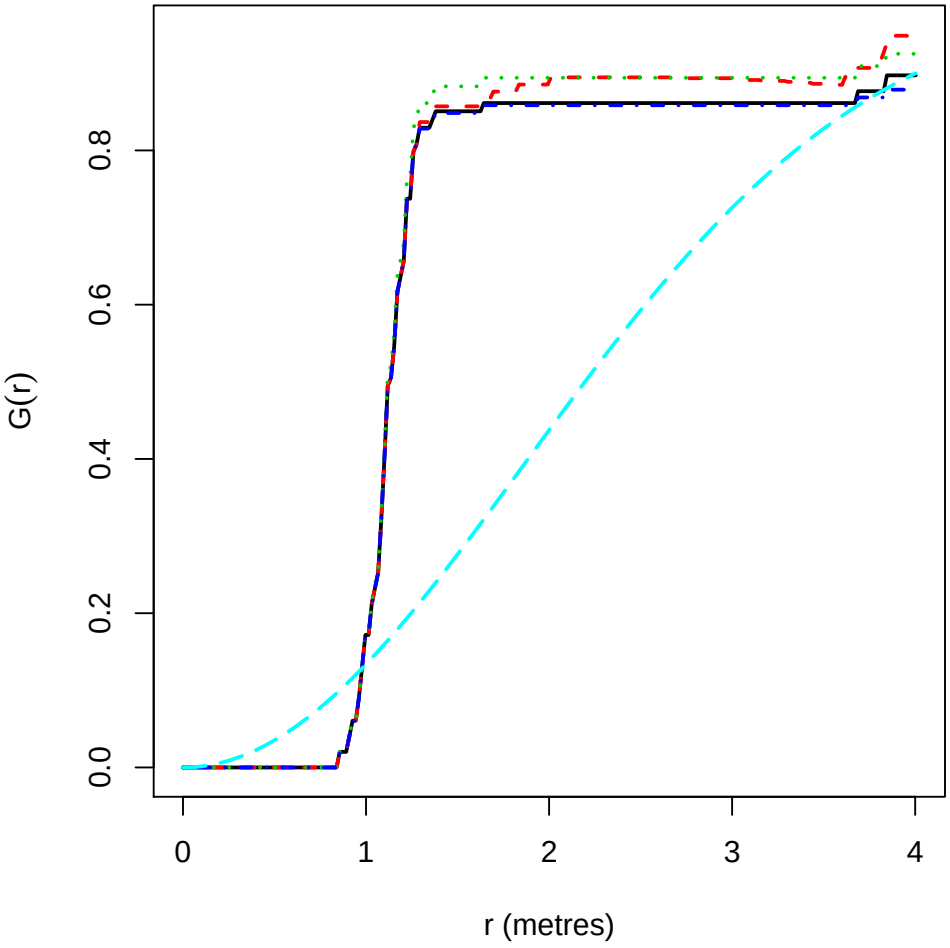
```
> Plot(pcfA, ylim=c(0,10))
```




```
> Plot(KA)
```




```
> Plot(GA)
```



4 Quadrature scheme

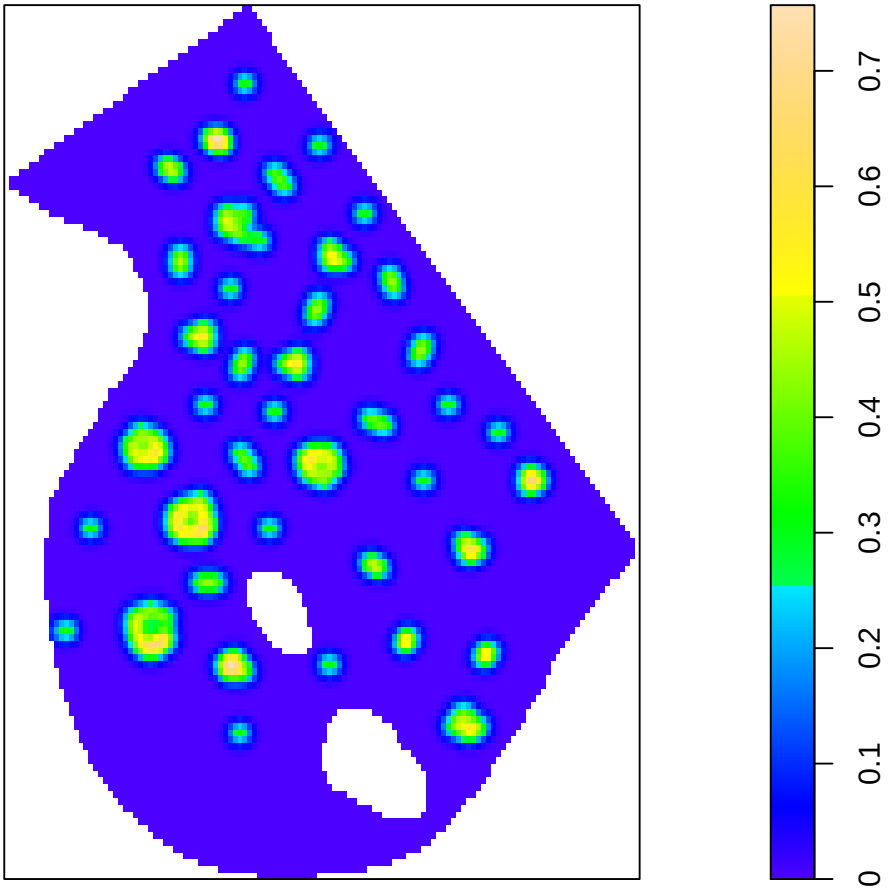
Make a quadrature scheme with spacing `eps` in each direction.

```
> gsquad <- quadscheme(gordon, eps=eps)
> gsquad

Quadrature scheme
99 data points,  8821 dummy points
Total weight  2164.41678053568
```

5 Inhomogeneity

First we show the result of a kernel smoothing estimate of point process intensity, with cross-validated bandwidth selection by the Berman-Diggle method.

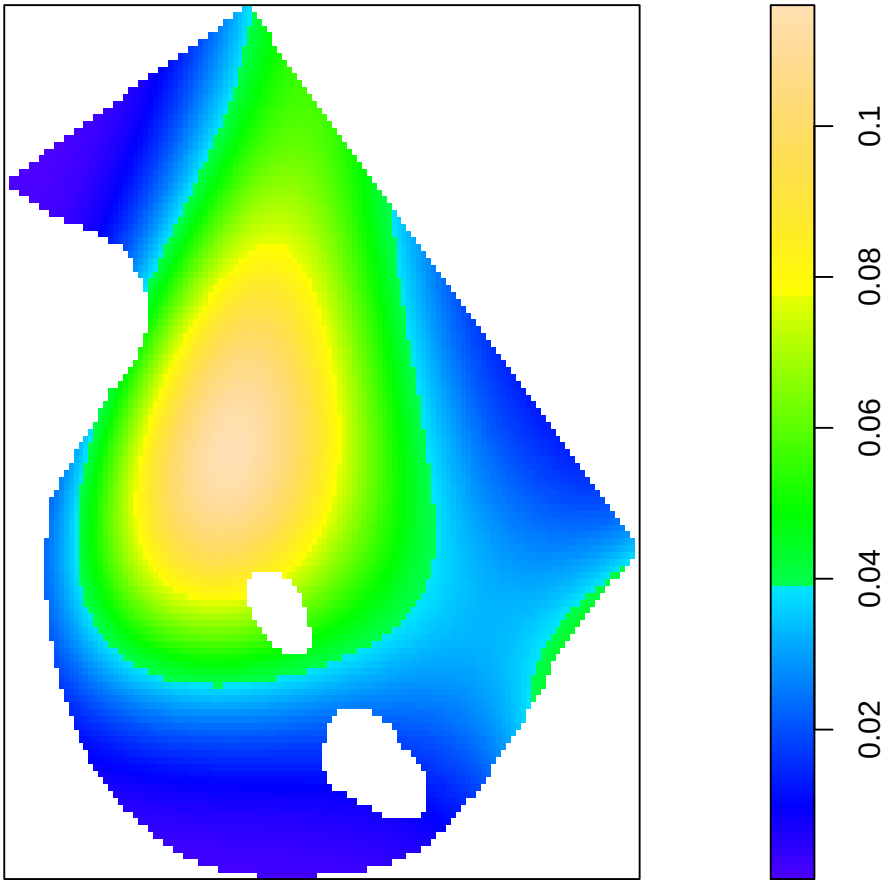
```
> Plot(density(gordon, bw.diggle, dimyx=NGRID))
```



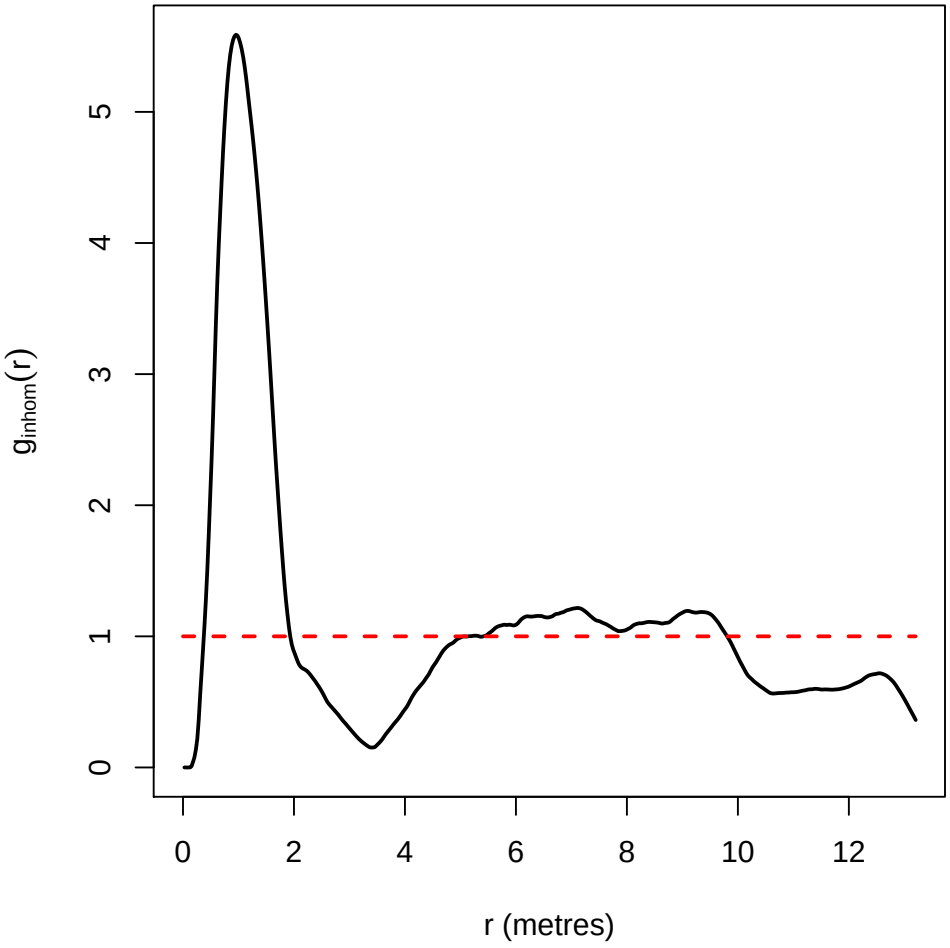
The chosen bandwidth appears to be too small to reveal a gradual spatial trend. The Berman-Diggle method was originally designed for Cox processes, and generally performs well for patterns that show positive association between points; apparently the inhibition between groups of sitters in this dataset has caused underestimation of the bandwidth.

To investigate gradual spatial trend, we fit a log-cubic Poisson model

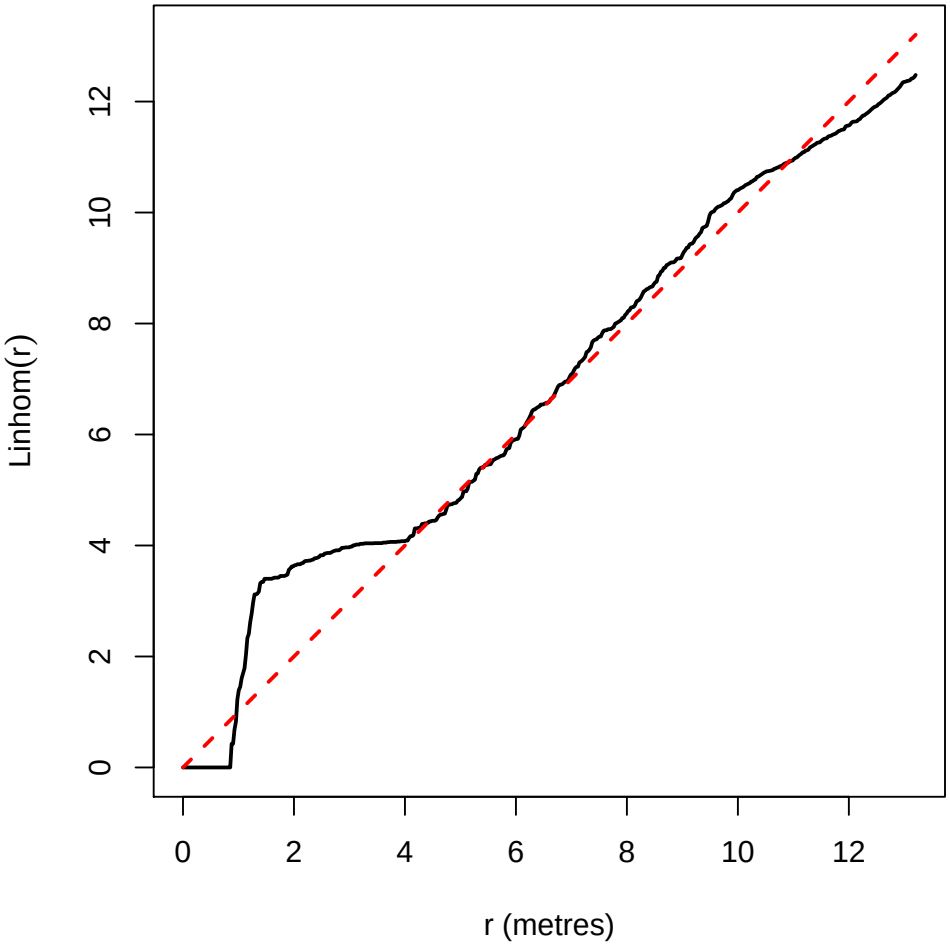
```
> fitcubic <- ppm(gsquad, ~polynom(x,y,3))
> predcubic <- predict(fitcubic, ngrid=NGRID)

> Plot(predcubic)
```




```
> pcfcubic <- pcfinhom(gordon, lambda=predcubic, nsim=39)
> Lcubic <- Linhom(gordon, lambda=predcubic)

> Plot(pcfcubic, cbind(iso, theo) ~ r)
```




```
> Plot(Lcubic, cbind(iso, theo) ~ r)
```

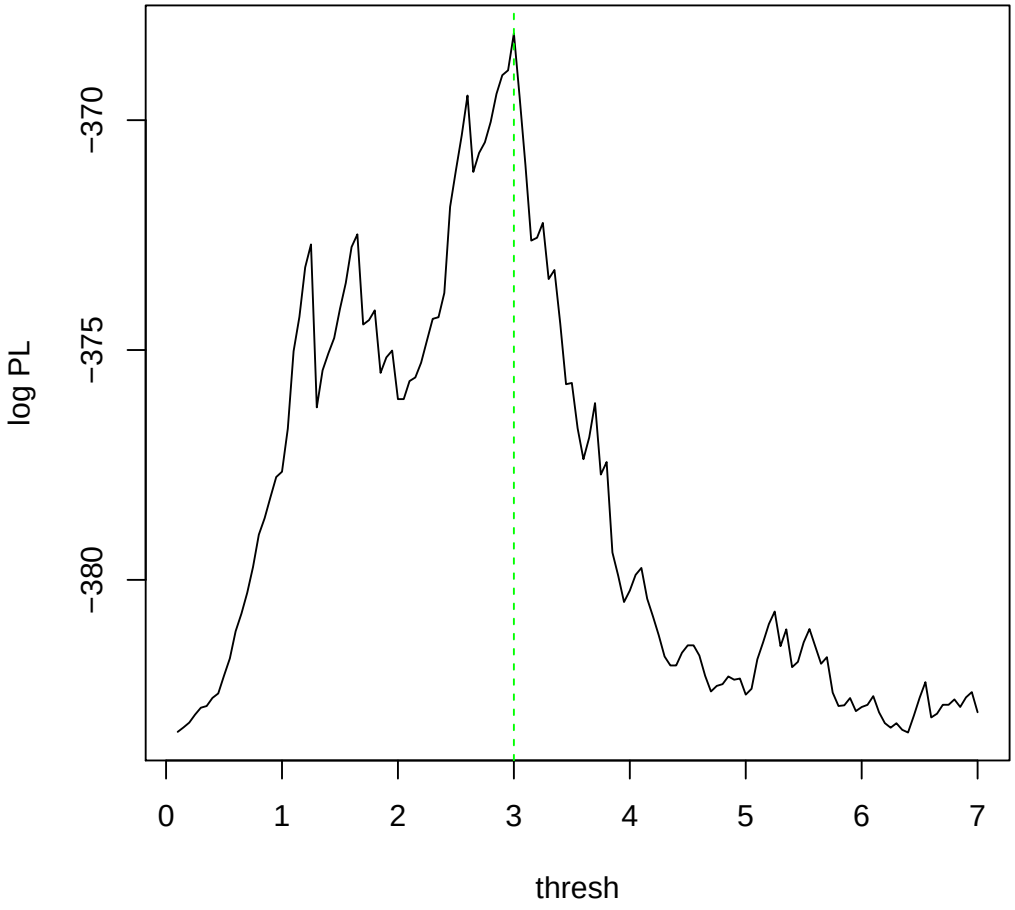


Comparison of the simulation and the original data suggests that there is a tendency in the Gordon Square data for people to avoid sitting close to the edge of the grassy area.

In the next calculation we fit a model that includes a term for proximity to the boundary of the window. This is a changepoint term which is constant above and below a threshold distance t from the boundary.

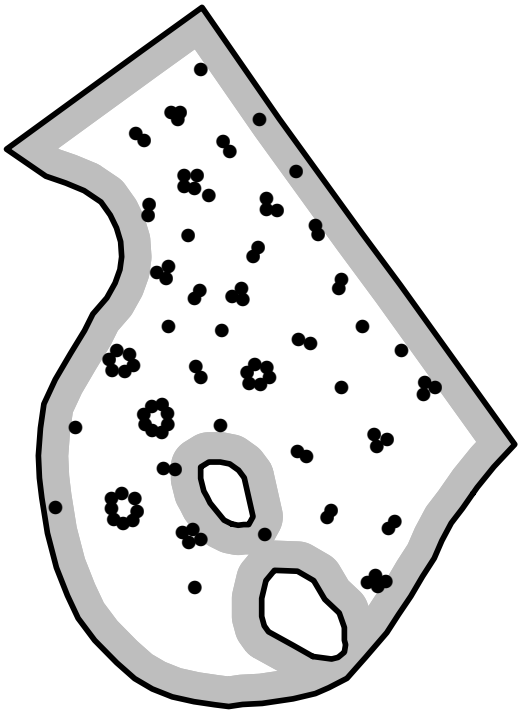
```
> b <- function(x,y,thresh=1) {inside.owin(x,y,border(as.owin(gordon), thresh))}
> borderprofile <- profilepl(s=data.frame(thresh=seq(0.1,7,by=0.05)),
+                            f=Poisson,
+                            gordon, ~polynom(x/25, y/25,3) + b,
+                            covariates=list(b=b),
+                            verbose=FALSE)
> borderprofile

Profile log pseudolikelihood values
for model:        ppm(gordon, ~polynom(x/25, y/25, 3) + b, interaction = Poisson,
    covariates = list(b = b))
fitted with rbord= 0
Interaction: Poisson
 with irregular parameter thresh in [0.1, 7]
Optimum value of irregular parameter: thresh = 3
```

```
> as.ppm(borderprofile)

Nonstationary Poisson process

Trend formula: ~polynom(x/25, y/25, 3) + b

Fitted coefficients for trend formula:
                            (Intercept)          polynom(x/25, y/25, 3)[(x/25)]
                             -2.6810193                             -1.8486919
         polynom(x/25, y/25, 3)[(y/25)]        polynom(x/25, y/25, 3)[(x/25)^2]
                             -0.1133362                              0.1487247
  polynom(x/25, y/25, 3)[(x/25).(y/25)]        polynom(x/25, y/25, 3)[(y/25)^2]
                              1.1319089                              0.2277235
       polynom(x/25, y/25, 3)[(x/25)^3] polynom(x/25, y/25, 3)[(x/25)^2.(y/25)]
                              2.4455168                              0.5643029
polynom(x/25, y/25, 3)[(x/25).(y/25)^2]        polynom(x/25, y/25, 3)[(y/25)^3]
                              3.9006704                              0.7952172
                                  bTRUE
                             -2.2619886

Covariate function arguments (covfunargs) provided:
thresh = 3
                                        Estimate      S.E. Ztest    CI95.lo
(Intercept)                           -2.6810193 0.2035577    na -3.0799851
polynom(x/25, y/25, 3)[(x/25)]        -1.8486919 0.6282875    ** -3.0801128
polynom(x/25, y/25, 3)[(y/25)]        -0.1133362 0.4476204       -0.9906562
polynom(x/25, y/25, 3)[(x/25)^2]       0.1487247 1.0235755       -1.8574464
polynom(x/25, y/25, 3)[(x/25).(y/25)]  1.1319089 1.0458757       -0.9179697
polynom(x/25, y/25, 3)[(y/25)^2]       0.2277235 0.4747658       -0.7028003
polynom(x/25, y/25, 3)[(x/25)^3]       2.4455168 1.4993041       -0.4930653
polynom(x/25, y/25, 3)[(x/25)^2.(y/25)] 0.5643029 2.2501999       -3.8460079
polynom(x/25, y/25, 3)[(x/25).(y/25)^2] 3.9006704 1.4077885    **  1.1414556
polynom(x/25, y/25, 3)[(y/25)^3]       0.7952172 0.4721694       -0.1302177
bTRUE                                 -2.2619886 0.5118113   *** -3.2651203
                                        CI95.hi
(Intercept)                           -2.2820534
polynom(x/25, y/25, 3)[(x/25)]        -0.6172709
polynom(x/25, y/25, 3)[(y/25)]         0.7639837
polynom(x/25, y/25, 3)[(x/25)^2]       2.1548958
polynom(x/25, y/25, 3)[(x/25).(y/25)]  3.1817875
polynom(x/25, y/25, 3)[(y/25)^2]       1.1582473
polynom(x/25, y/25, 3)[(x/25)^3]       5.3840989
polynom(x/25, y/25, 3)[(x/25)^2.(y/25)] 4.9746137
polynom(x/25, y/25, 3)[(x/25).(y/25)^2] 6.6598851
polynom(x/25, y/25, 3)[(y/25)^3]       1.7206521
bTRUE                                 -1.2588569

> plot(borderprofile, main="")
```



The selected threshold is 3 metres, confirming our visual estimate.

```
> gwin <- as.owin(gordon)
> b3 <- border(gwin, 3)
> plot(b3, main="", col="grey", border="grey")
> plot(gwin, add=TRUE, lwd=3)
> plot(gordon, add=TRUE, pch=16)
```



6 Hybrid modelling

6.1 No edge correction

The "standard model" for point process analysis assumes that the point process exists on the whole Euclidean plane, but is only observed through a bounded window. Edge effects intrude into the analysis because a point near the edge of the observable window might have an unseen neighbour just outside the window. Edge corrections are often advisable in this case.

The "standard model" is inappropriate for the Gordon Square data because people cannot or do not sit outside the boundary of the square. This suggests that we model the data as a finite point process living inside the given boundary, and correspondingly **avoid** edge correction in the exploratory analysis. When models need to be fitted, we will set `rbord=0` so that no edge correction is performed.

6.2 First model

There should be a hard core at short distances.

```
> n <- npoints(gordon)
> r0 <- min(nndist(gordon)) * n/(n+1)
```

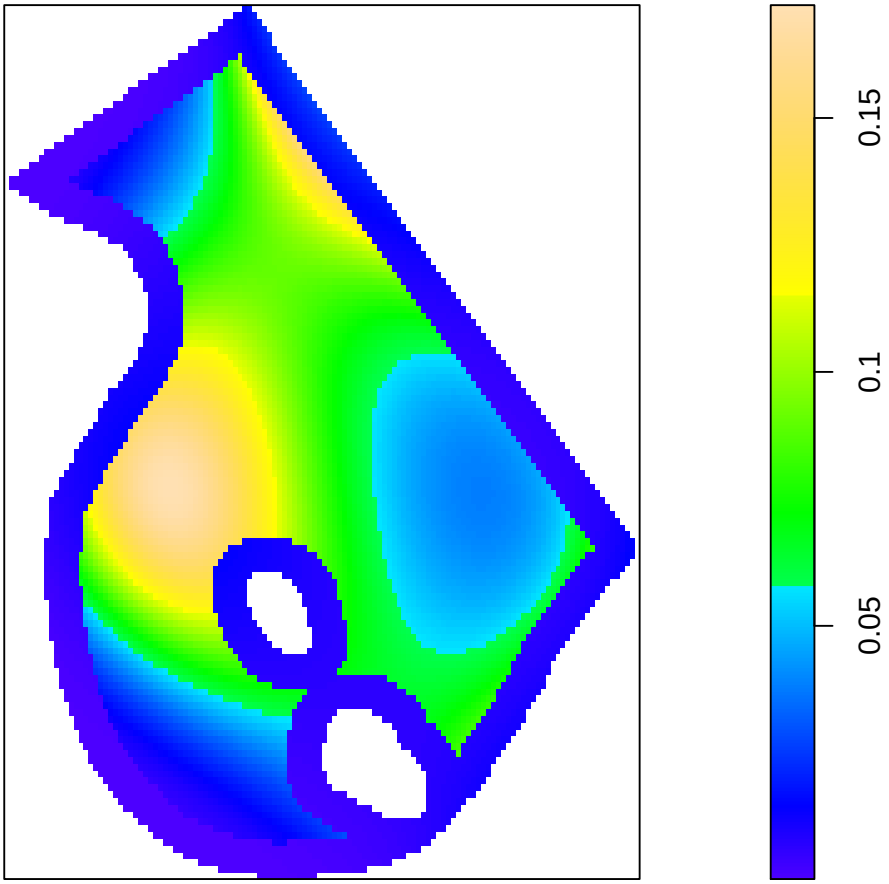
```
> r0 <- round(r0, 2)
> r0

[1] 0.85
```

Fit the hard core model with log-cubic trend, without edge correction.

```
> fit1 <- ppm(gsquad, ~b + polynom(x,y,3), Hardcore(r0),
+             covariates=list(b=b3), rbord=0)
> fit1

Nonstationary Hard core process

Trend formula: ~b + polynom(x, y, 3)

Fitted coefficients for trend formula:
           (Intercept)                    bTRUE     polynom(x, y, 3)[x]
         -2.520657e+00          -2.323792e+00          -8.346800e-02
    polynom(x, y, 3)[y]   polynom(x, y, 3)[x^2]   polynom(x, y, 3)[x.y]
         -8.678318e-03           7.386439e-05           2.072750e-03
  polynom(x, y, 3)[y^2]   polynom(x, y, 3)[x^3] polynom(x, y, 3)[x^2.y]
          3.960424e-04           1.749966e-04           6.167654e-05
polynom(x, y, 3)[x.y^2]   polynom(x, y, 3)[y^3]
          2.709802e-04           5.704575e-05

Interaction: Hard core process
hard core distance:        0.85

For standard errors, type coef(summary(x))
```

Intensity:

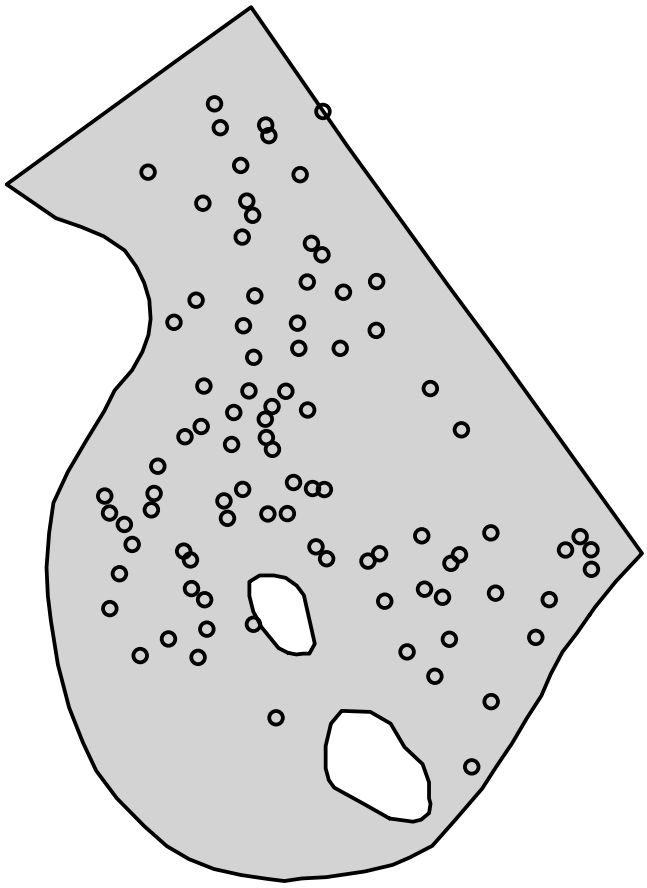
```
> Plot(predict(fit1))
```



Here is a simulated realisation from the model. Note that we specify 'no expansion' of the simulation window (although this is automatic in spatstat 1.24-2 and later) to avoid some strange behaviour (and to be consistent with treating the boundary of Gordon Square as 'hard').

```
> sim1 <- simulate(fit1, expand=1, nrep=Nrmh)[[1]]
```

```
> PlotGordon(sim1)
```



Count the number of points

```
> npoints(sim1)
```

```
[1] 90
```

Compute the residual diagnostics for this model.

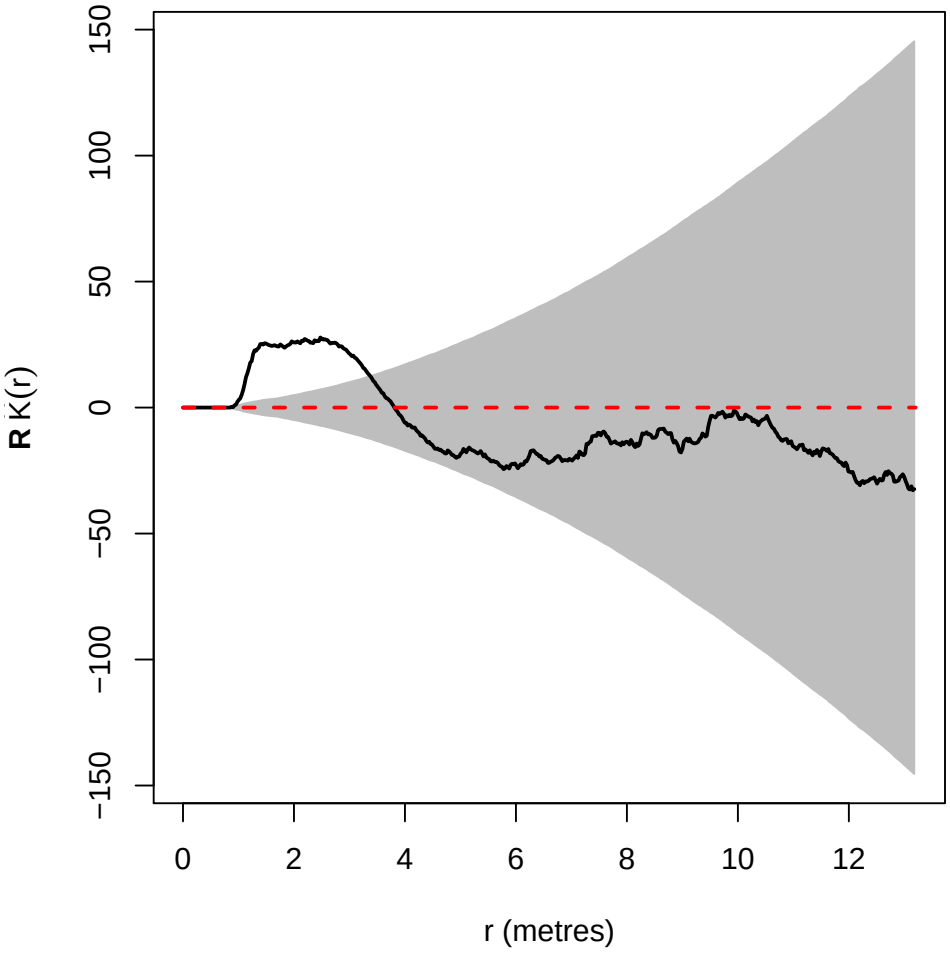
```
> if(!file.exists("fit1residuals.rda") || recompute) {
+   k1 <- Kres(fit1, correction="best")
+   g1 <- Gres(fit1, correction="best")
+   a1 <- psstA(fit1)
+   t1 <- psst(fit1, Tstat,
+              funargs=list(correction="best", verbose=FALSE),
+              verbose=FALSE)
+   save(k1, g1, a1, t1, file="fit1residuals.rda", compress=TRUE)
+ } else load("fit1residuals.rda")
```

Note that, although we are not using edge correction when fitting the model, these residuals are valid for any value of the argument `correction` (which only determines the functional that enters into the residual).

```
> Kshade <- c("ihi", "ilo")
> Gshade <- Tshade <- c("hi", "lo")
> Ashade <- c("up", "lo")
```

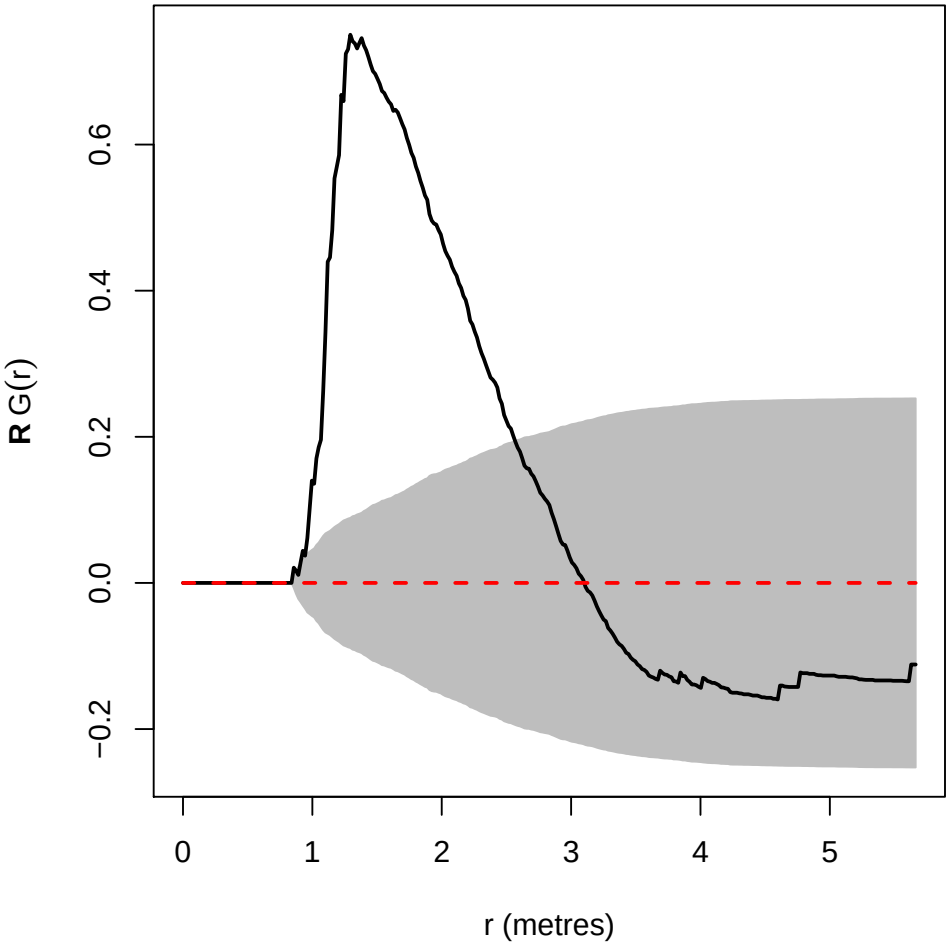
Here is the K residual for this fitted model.

```
> Plot(k1, shade=Kshade)
```



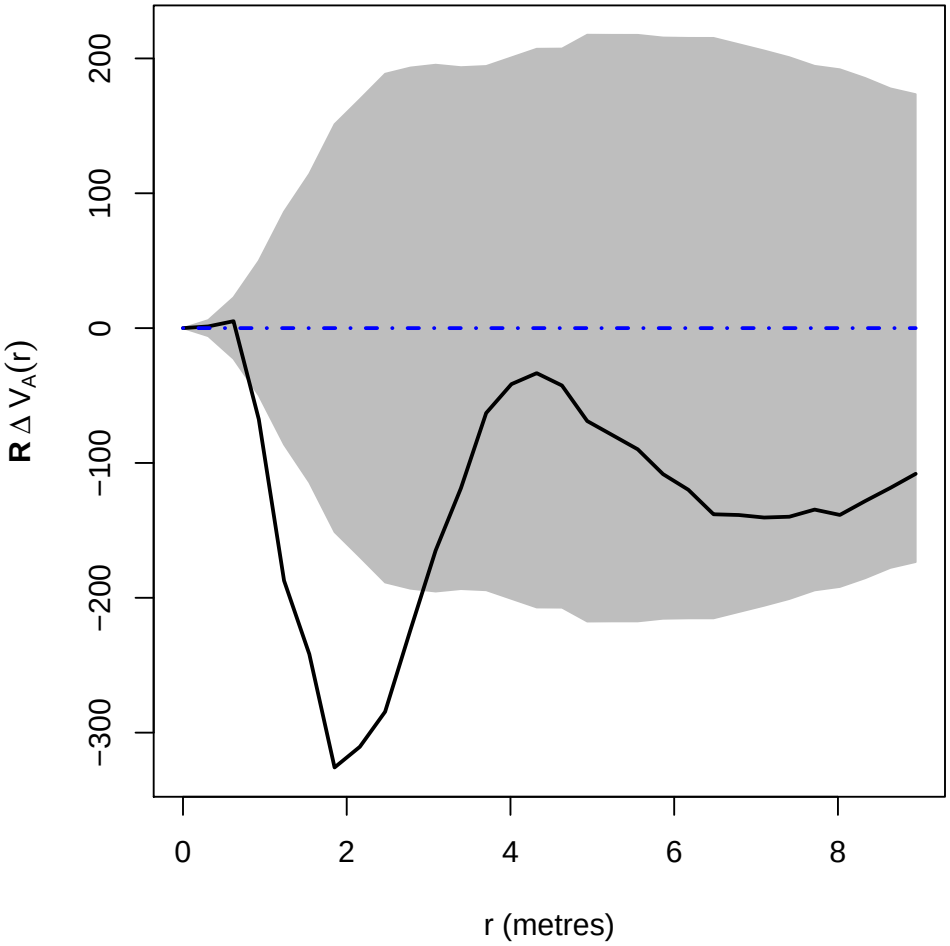
Here is the G residual (which has a similar interpretation):

```
> Plot(g1, shade=Gshade)
```



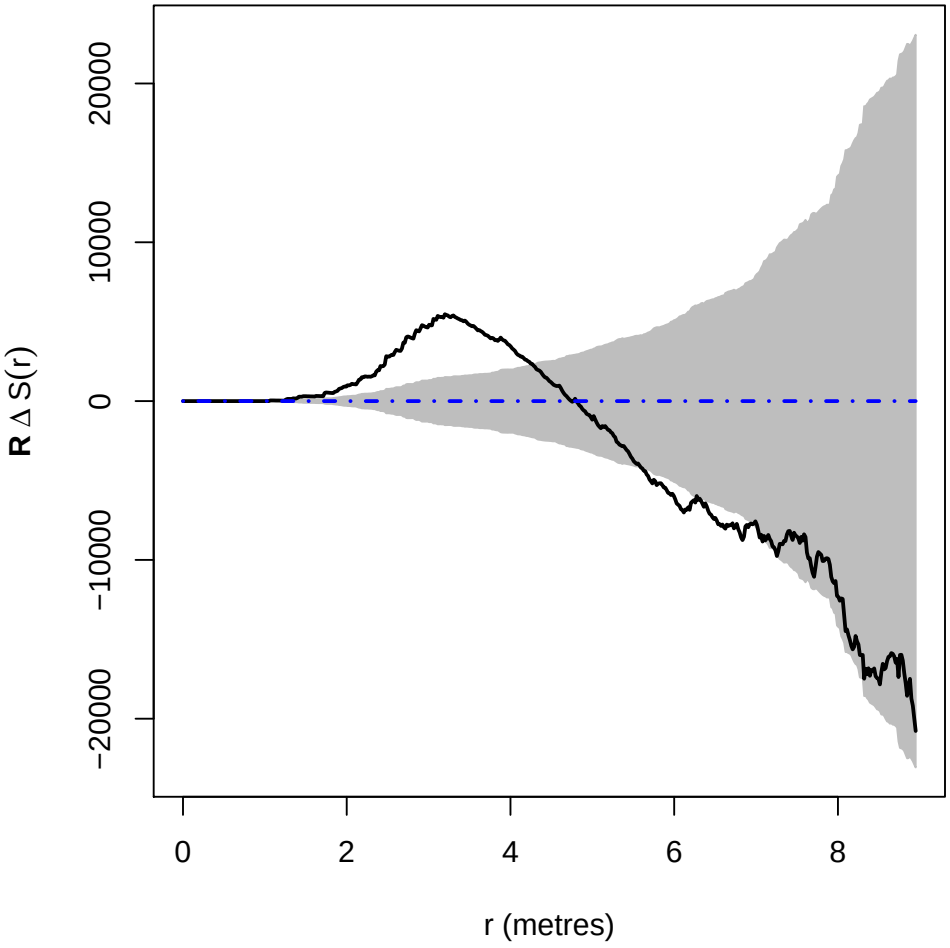
Here is the area pseudo-residual (which has a similar interpretation, related to the empty space function F with a few differences):

```
> Plot(a1, shade=Ashade)
```



Here is the triplets pseudo-residual (related to the triplets summary function):

```
> Plot(t1, shade=Tshade)
```



6.3 Second model

The residual plots above suggest that, after allowing for the hard core, there is positive association which peaks at a distance of about 1.35. The sharpness of the G function peak suggests a Geyer saturation interaction.

Let's now fit a hybrid interaction consisting of the previously fitted model plus a Geyer interaction with parameters selected by maximum profile pseudolikelihood.

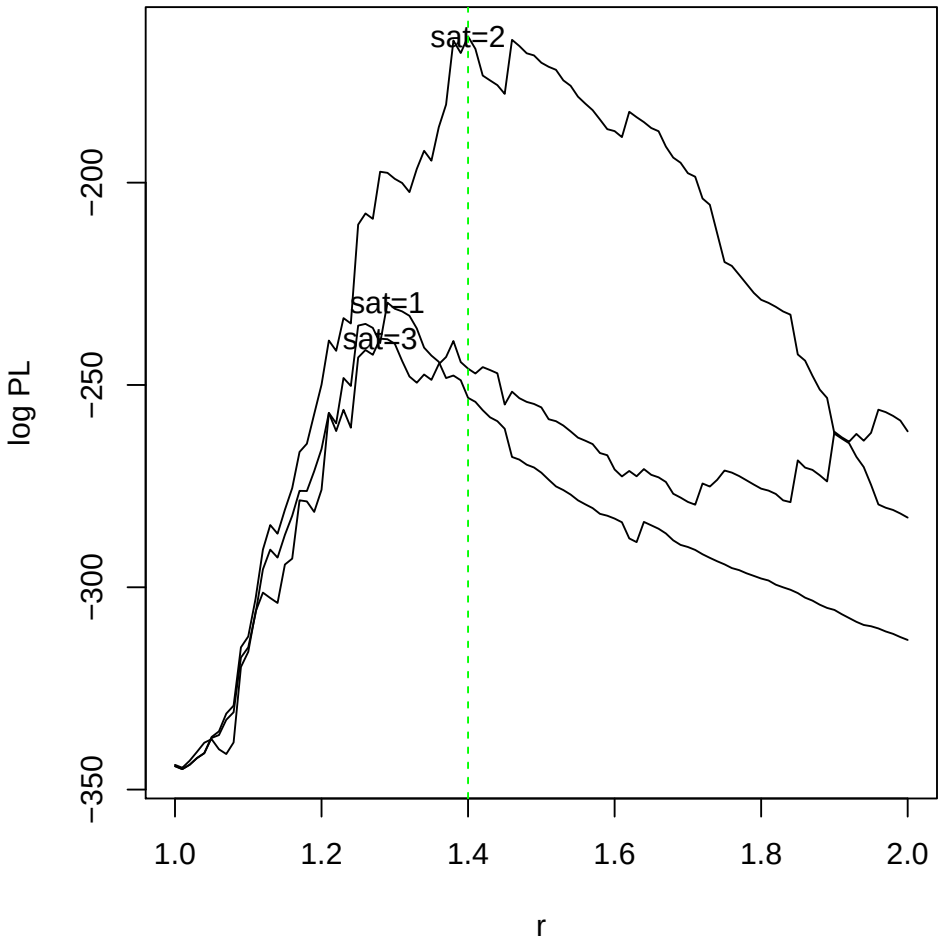
Note: I initially searched over wider ranges of the parameters $1 \leq r \leq 4$ and $s = 1, 2, 3, 4$ to identify the optimum.

I have to rescale the x and y coordinates to prevent numerical instability.

```
> hybridHardGeyer <- function(r, sat) {
+   Hybrid(H=Hardcore(r0), G=Geyer(r, sat))
+ }
> df <- expand.grid(r=seq(1,2,by=0.01), sat=1:3)
> fit2p <- profilepl(df, hybridHardGeyer,
+                    gsquad, ~b + polynom(x/25,y/25,3),
+                    covariates=list(b=b3),
+                    rbord=0,
+                    verbose=FALSE)
```

Profile pseudolikelihood plot:

```
> Plot(fit2p, lwd=2)
```

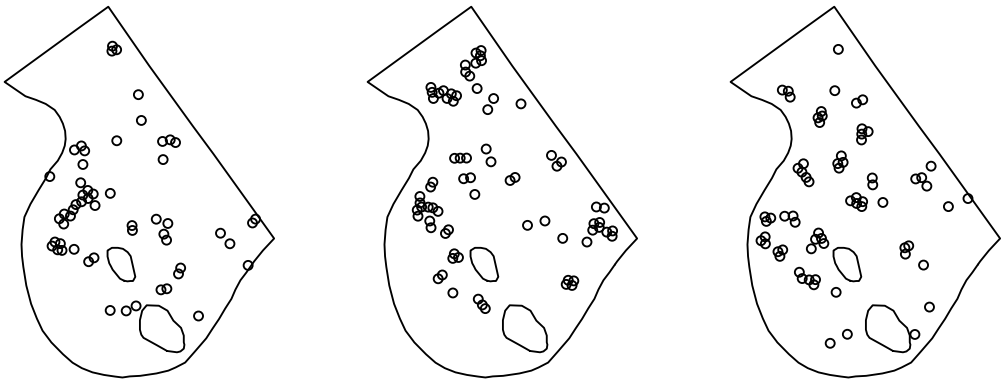



```
> fit2 <- as.ppm(fit2p)
> fit2

Nonstationary Hybrid interaction

Trend formula: ~b + polynom(x/25, y/25, 3)

Fitted coefficients for trend formula:
                                (Intercept)                                                bTRUE
                                -4.54105397                                          -1.24931909
          polynom(x/25, y/25, 3)[(x/25)]          polynom(x/25, y/25, 3)[(y/25)]
                                -0.82566827                                          -0.01902361
        polynom(x/25, y/25, 3)[(x/25)^2]   polynom(x/25, y/25, 3)[(x/25).(y/25)]
                                 1.04363851                                           1.07779098
        polynom(x/25, y/25, 3)[(y/25)^2]        polynom(x/25, y/25, 3)[(x/25)^3]
                                 0.44770965                                           0.96431690
polynom(x/25, y/25, 3)[(x/25)^2.(y/25)] polynom(x/25, y/25, 3)[(x/25).(y/25)^2]
```

```
                            1.20689235                                  2.46988612
       polynom(x/25, y/25, 3)[(y/25)^3]
                            0.40664594

Interaction: Hybrid of 2 components: H and G

H:
Interaction:Hard core process
hard core distance:        0.85
G:
Interaction:Geyer saturation process
interaction distance:        1.4
 saturation parameter:         2

Fitted G interaction parameter gamma:        3.9489

Relevant coefficients:
      G.
1.373429

For standard errors, type coef(summary(x))
```

Here are 3 simulated realisations from the model:

```
> sim2 <- simulate(fit2, nsim=3, expand=1, nrep=Nrmh)

Generating 3 simulated patterns ...1, 2,  3.

> Plot(sim2, main.panel="")
```



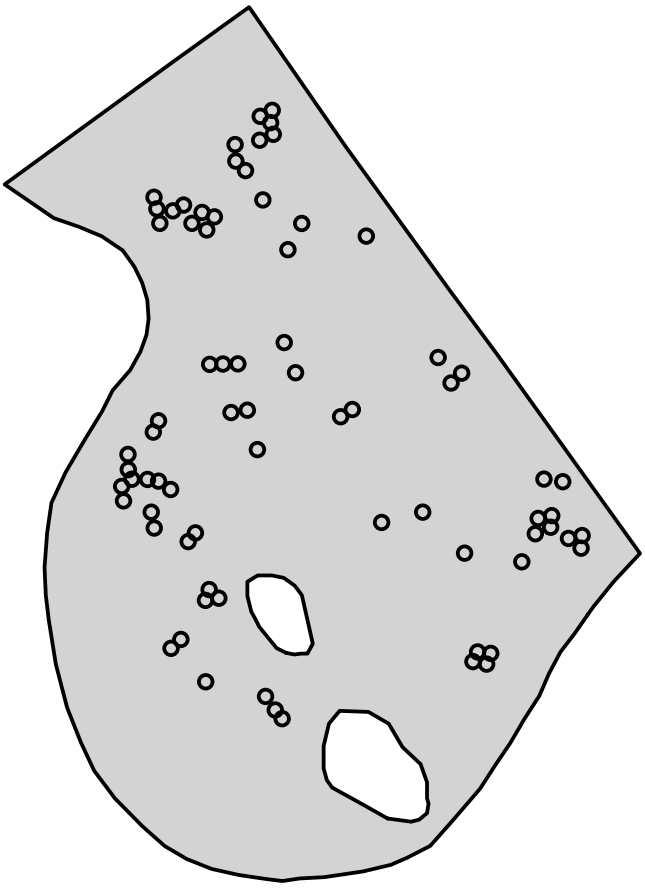
Count the numbers of points

```
> unlist(lapply(sim2, npoints))
```

```
Simulation 1 Simulation 2 Simulation 3
          56           74           69
```

For better comparison with the data, here is a plot of one realisation.

```
> PlotGordon(sim2[[2]])
```

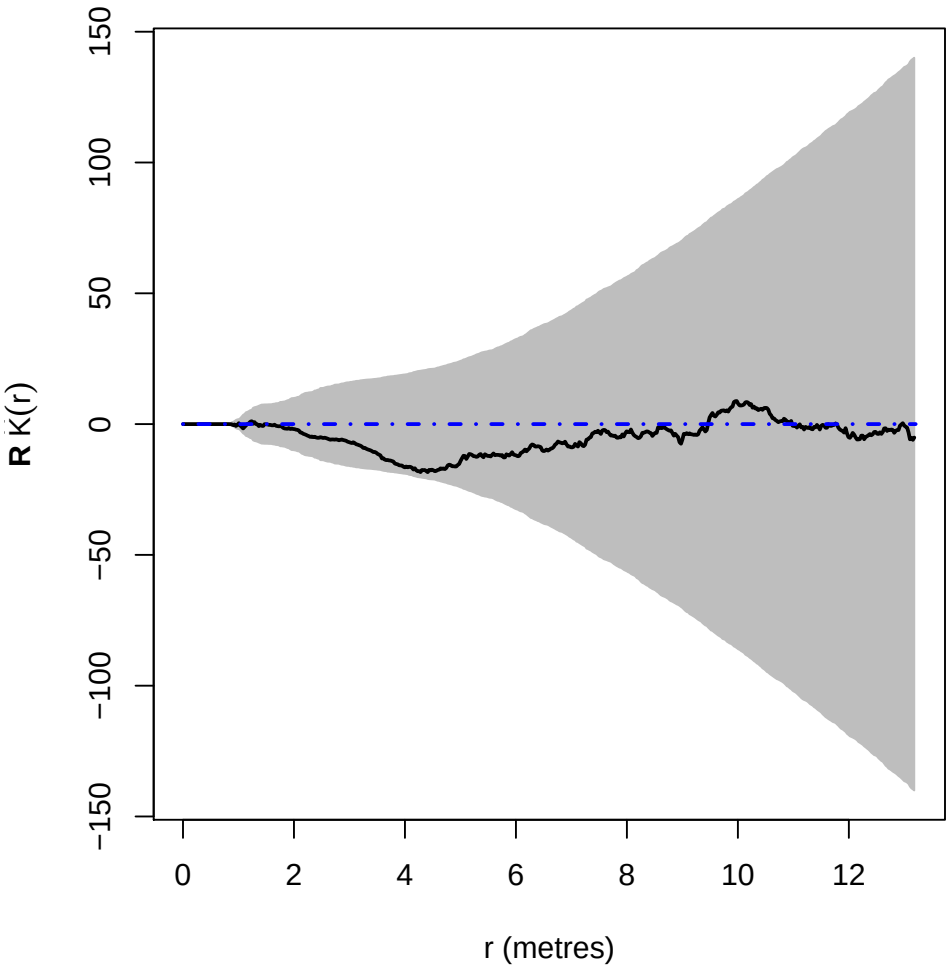


Compute the residual diagnostics for this model.

```
> if(!file.exists("fit2residuals.rda") || recompute) {
+   k2 <- Kres(fit2, correction="best")
+   g2 <- Gres(fit2, correction="best")
+   a2 <- psstA(fit2)
+   t2 <- psst(fit2, Tstat,
+              funargs=list(correction="best", verbose=FALSE),
+              verbose=FALSE)
+   save(k2, g2, a2, t2, file="fit2residuals.rda", compress=TRUE)
+ } else load("fit2residuals.rda")
```

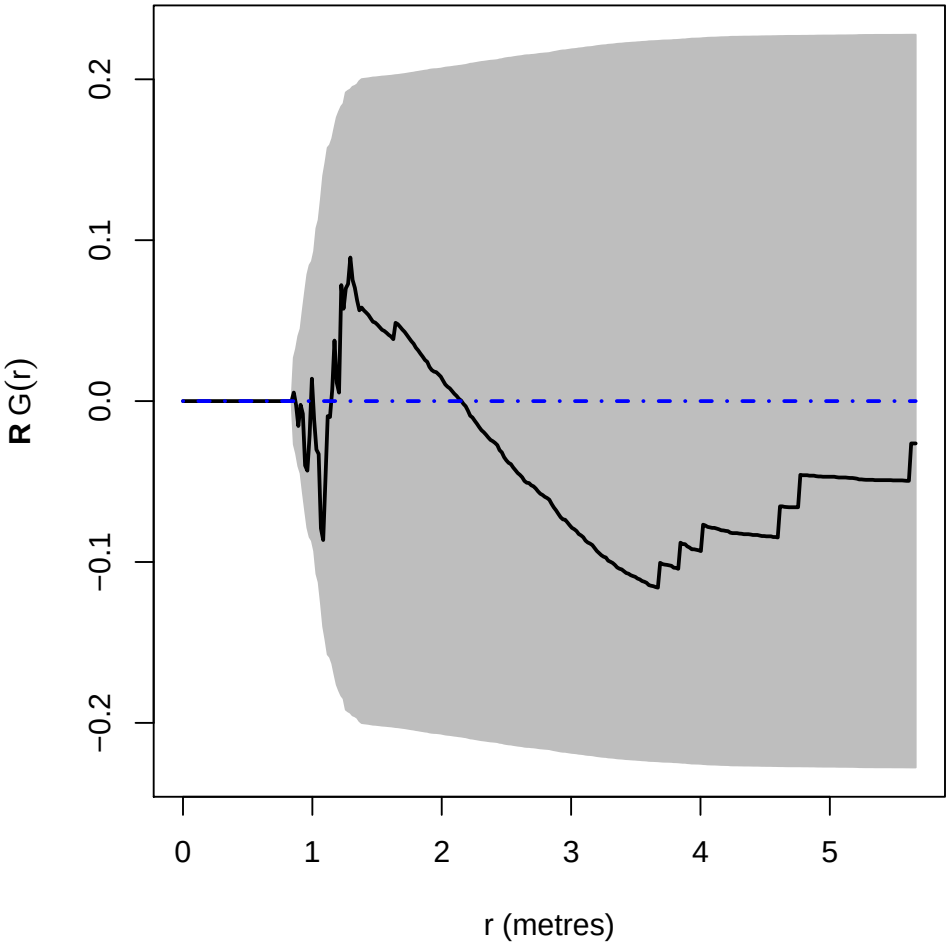
Here is the K residual (reweighted isotropic correction):

```
> Plot(k2, cbind(ires, ihi, ilo, theo) ~ r, shade=Kshade)
```



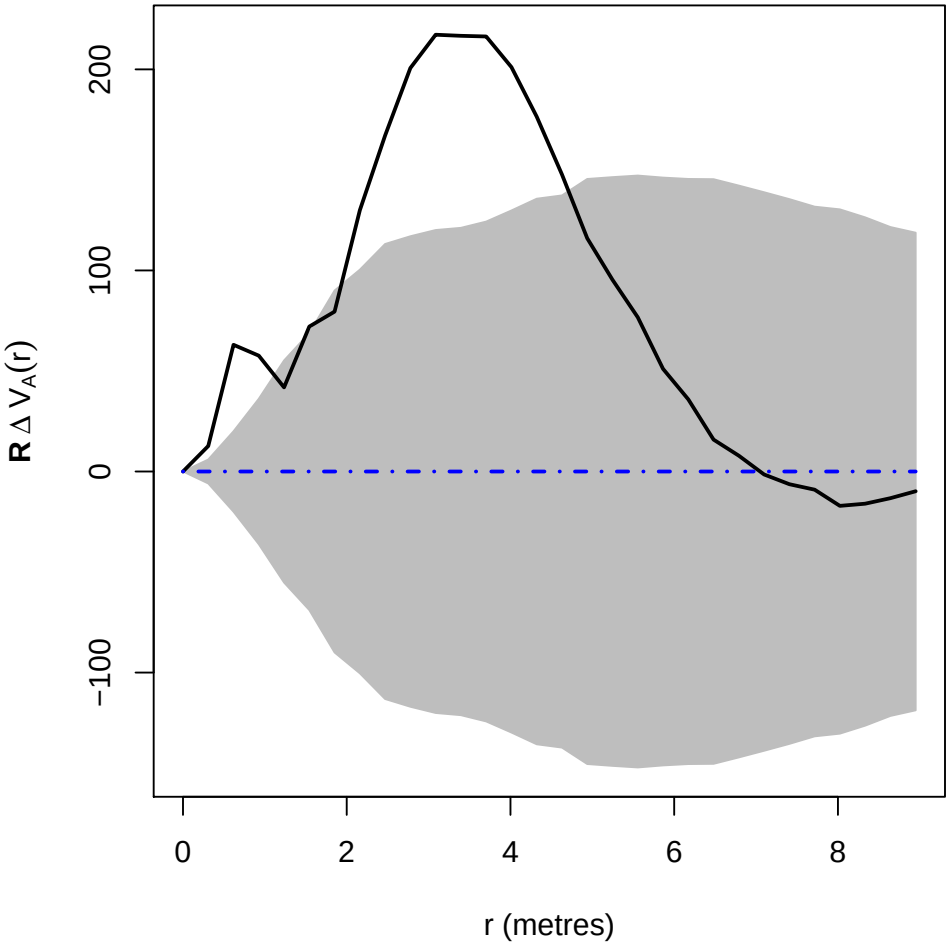
Here is the G residual:

```
> Plot(g2, legendpos="topright", cbind(hres, hi, lo, theo) ~ r, shade=Gshade)
```



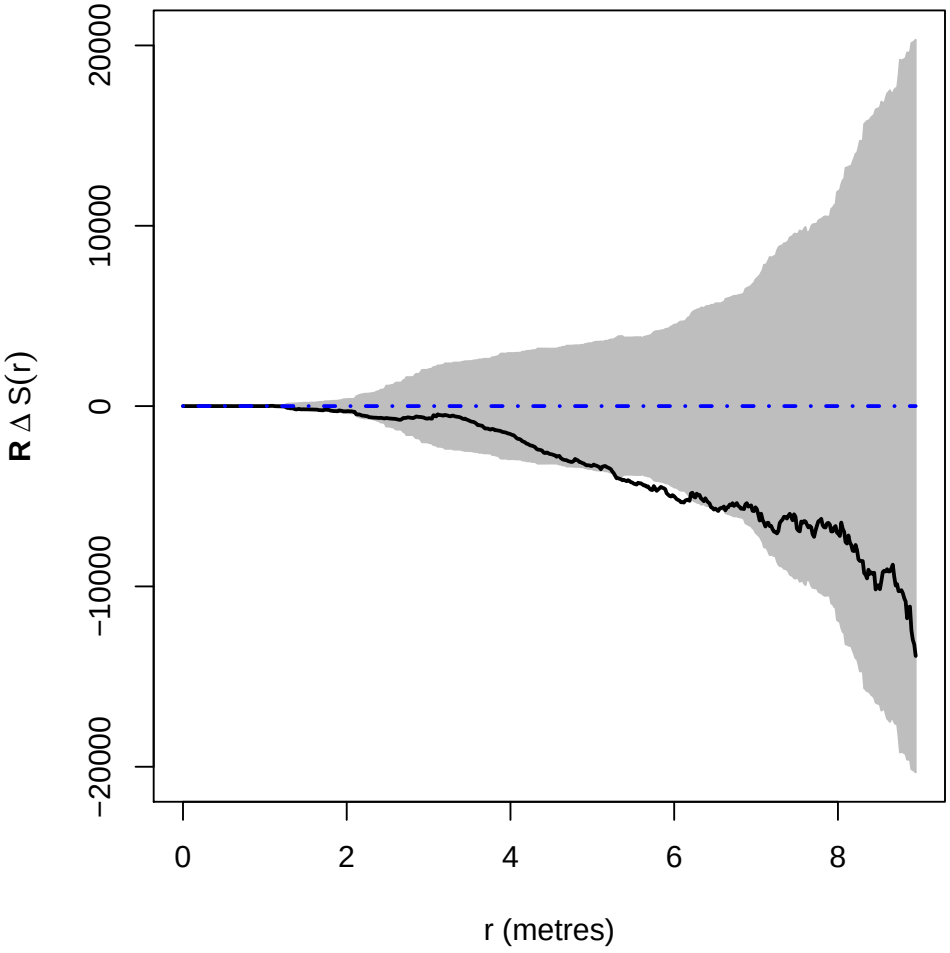
Here is the area pseudo-residual:

```
> Plot(a2, shade=Ashade)
```



Here is the triplets pseudo-residual:

```
> Plot(t2, shade=Tshade)
```



6.4 Third model

The diagnostics for the second model suggest that a further negative association should be included. The erratic behaviour of the G residual for the second model suggests that there is some extra interaction at a distance of about 1.5 metres. The area residual, with a peak at about 3 metres, can also be interpreted as suggesting an extra interaction at a distance of 1.5 metres.

When the simulated realisations of the second model are compared with the original data, focusing on patterns a scale of about 1.5 metres, it is noticeable that the original data contain relatively few triangles of people at this short distance: most of the arrangements of people are circles of about 5 metres in diameter, in which each person is flanked by two close neighbours.

Accordingly an interesting possibility is to add a Triplets interaction component having the *same* interaction distance as the Geyer component (currently estimated as 1.4 metres). Suppose this model has interaction parameter $\gamma > 1$ for the Geyer component and $\zeta < 1$ for the Triplets component. Then the Geyer component encourages more 'sociable' arrangements where more people sit together, but with no extra benefit when a person has more than 2 neighbours; while the Strauss component penalises 'crowding'.

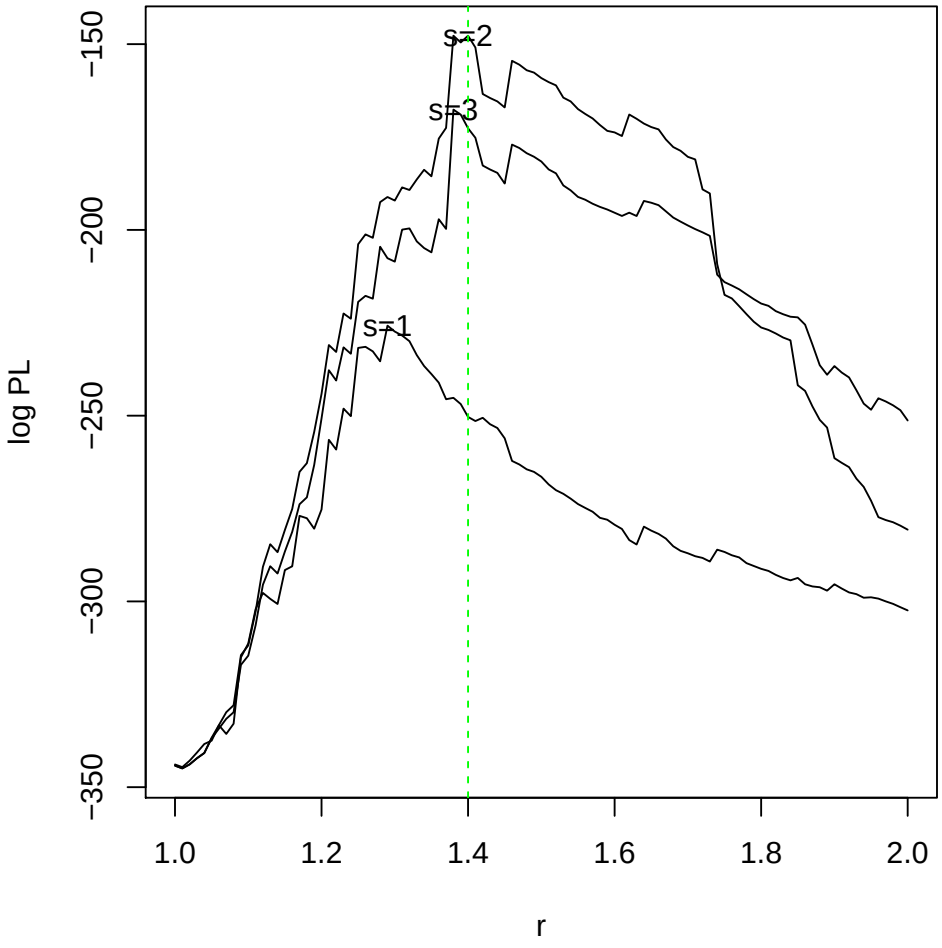
We'll select this by maximum profile pseudolikelihood as well. We'll search over all the irregular parameters, but narrow down the search range to a neighbourhood of the previously-fitted value.

```
> hybrid3 <- function(r, s) {
```

```
+   Hybrid(Hard=Hardcore(r0), Social=Geyer(r, s), Crowding=Triplets(r))
+ }
> df <- expand.grid(r=seq(1, 2, by=0.01),
+                   s=1:3)
> fit3p <- profilepl(df, hybrid3, gsquad,
+                    ~b + polynom(x/25,y/25,3),
+                    covariates=list(b=b3),
+                    rbord=0,
+                    verbose=FALSE)
```

Profile pseudolikelihood plot:

```
> Plot(fit3p, lwd=2)
```

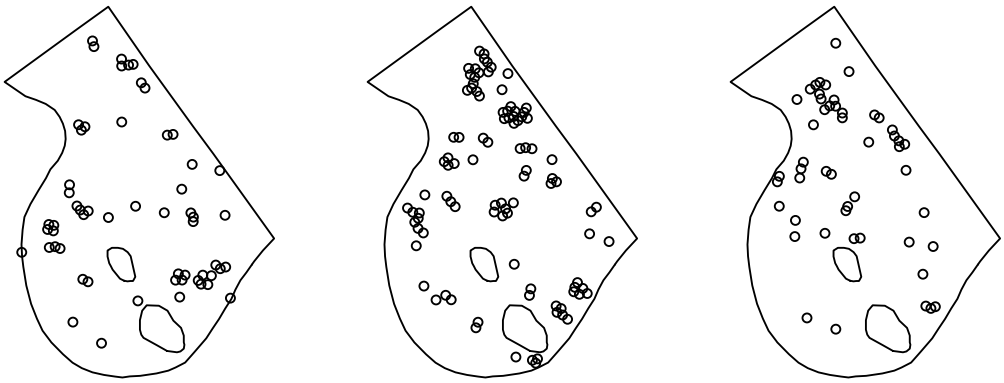



```
> fit3 <- as.ppm(fit3p)
> fit3

Nonstationary Hybrid interaction

Trend formula: ~b + polynom(x/25, y/25, 3)
```

```
Fitted coefficients for trend formula:
                            (Intercept)                                    bTRUE
                             -4.7284582                               -1.2999369
         polynom(x/25, y/25, 3)[(x/25)]          polynom(x/25, y/25, 3)[(y/25)]
                             -0.5595024                                0.2198046
       polynom(x/25, y/25, 3)[(x/25)^2]   polynom(x/25, y/25, 3)[(x/25).(y/25)]
                              0.7436236                                0.5363275
       polynom(x/25, y/25, 3)[(y/25)^2]        polynom(x/25, y/25, 3)[(x/25)^3]
                              0.5883816                                0.6692775
polynom(x/25, y/25, 3)[(x/25)^2.(y/25)] polynom(x/25, y/25, 3)[(x/25).(y/25)^2]
                              0.3940539                                2.3963390
       polynom(x/25, y/25, 3)[(y/25)^3]
                              0.1215170

Interaction: Hybrid of 3 components: Hard, Social and Crowding

Hard:
Interaction:Hard core process
hard core distance:        0.85
Social:
Interaction:Geyer saturation process
interaction distance:        1.4
 saturation parameter:         2
Crowding:
Interaction:Triplets process
interaction distance:        1.4

Fitted Social interaction parameter gamma:         4.9633
Fitted Crowding interaction parameter gamma:         0.2351

Relevant coefficients:
  Social. Crowding.
 1.602069 -1.447896

For standard errors, type coef(summary(x))
```

Here are 3 simulated realisations from the model:

```
> set.seed(99)
> sim3 <- simulate(fit3, nsim=3, expand=1, nrep=Nrmh)

Generating 3 simulated patterns ...1, 2,  3.

> Plot(sim3, main.panel="")
```



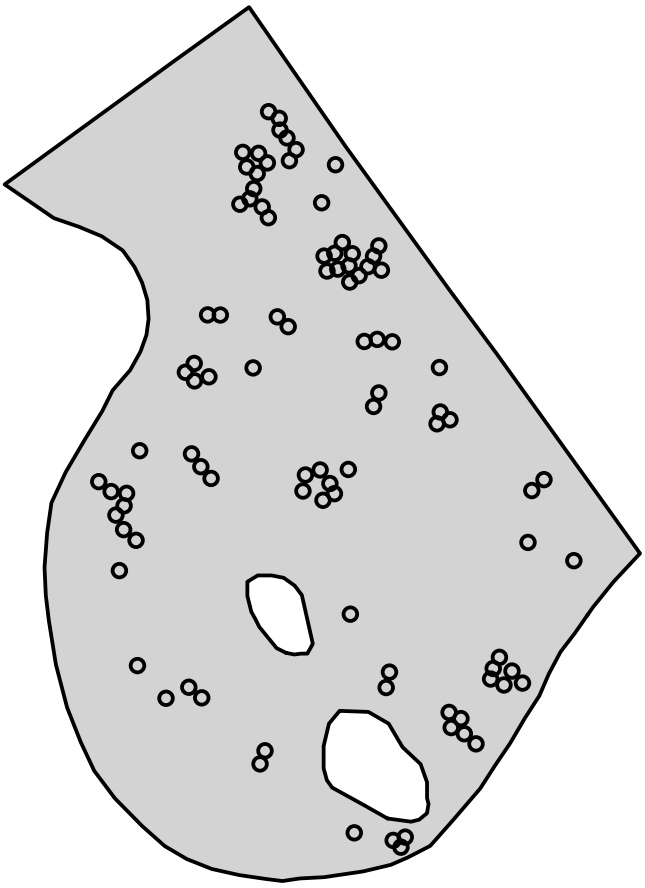
Count the numbers of points

```
> unlist(lapply(sim3, npoints))
```

```
Simulation 1 Simulation 2 Simulation 3
          57           96           50
```

For better comparison with the data, here is a plot of one realisation.

```
> PlotGordon(sim3[[2]])
```

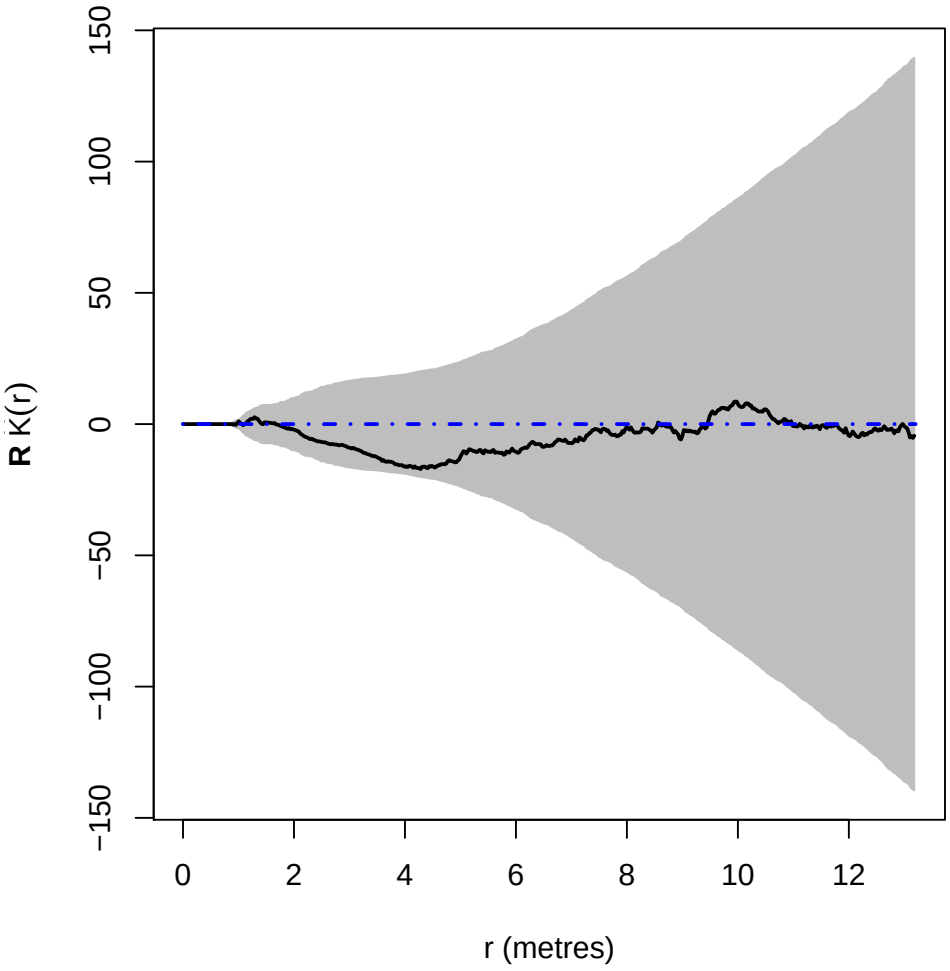


Compute the residual diagnostics for this model.

```
> if(!file.exists("fit3residuals.rda") || recompute) {
+   k3 <- Kres(fit3, correction="best")
+   g3 <- Gres(fit3, correction="best")
+   a3 <- psstA(fit3)
+   t3 <- psst(fit3, Tstat,
+              funargs=list(correction="best", verbose=FALSE),
+              verbose=FALSE)
+   save(k3, g3, a3, t3, file="fit3residuals.rda", compress=TRUE)
+ } else load("fit3residuals.rda")
```

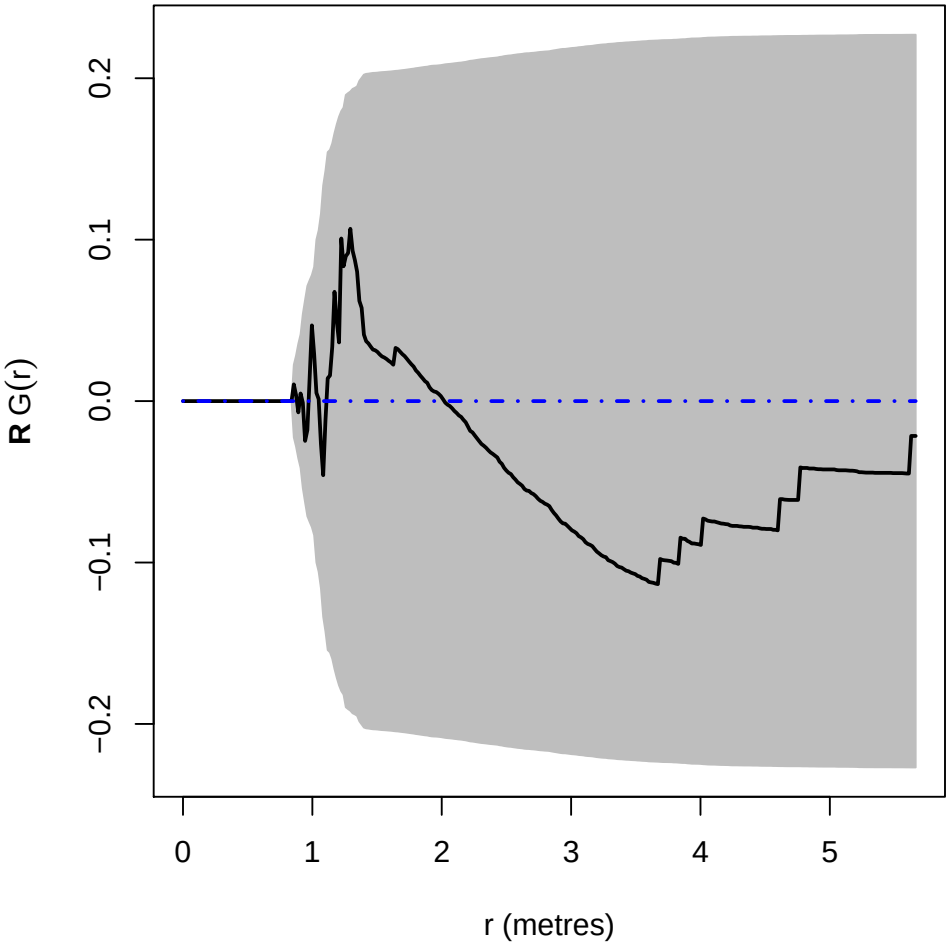
Here is the K residual:

```
> Plot(k3, cbind(ires, ihi, ilo, theo) ~ r, shade=Kshade)
```



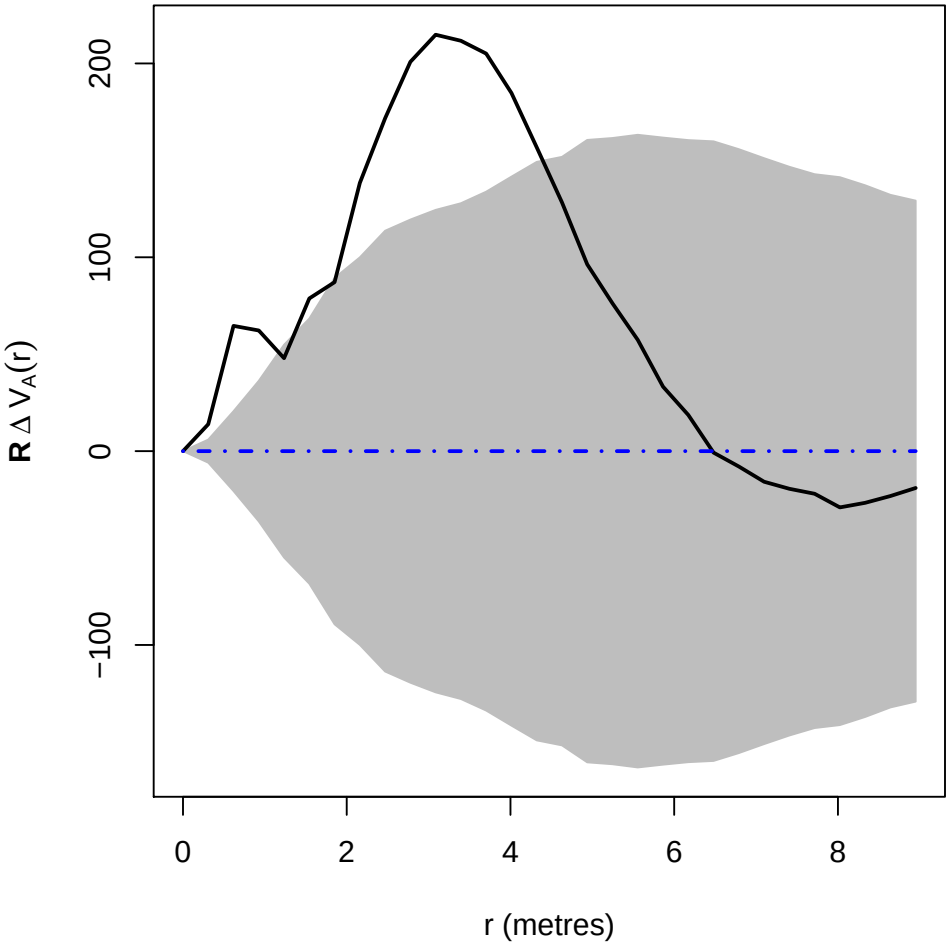
Here is the G residual:

```
> Plot(g3, legendpos="topright", cbind(hres, hi, lo, theo) ~ r, shade=Gshade)
```



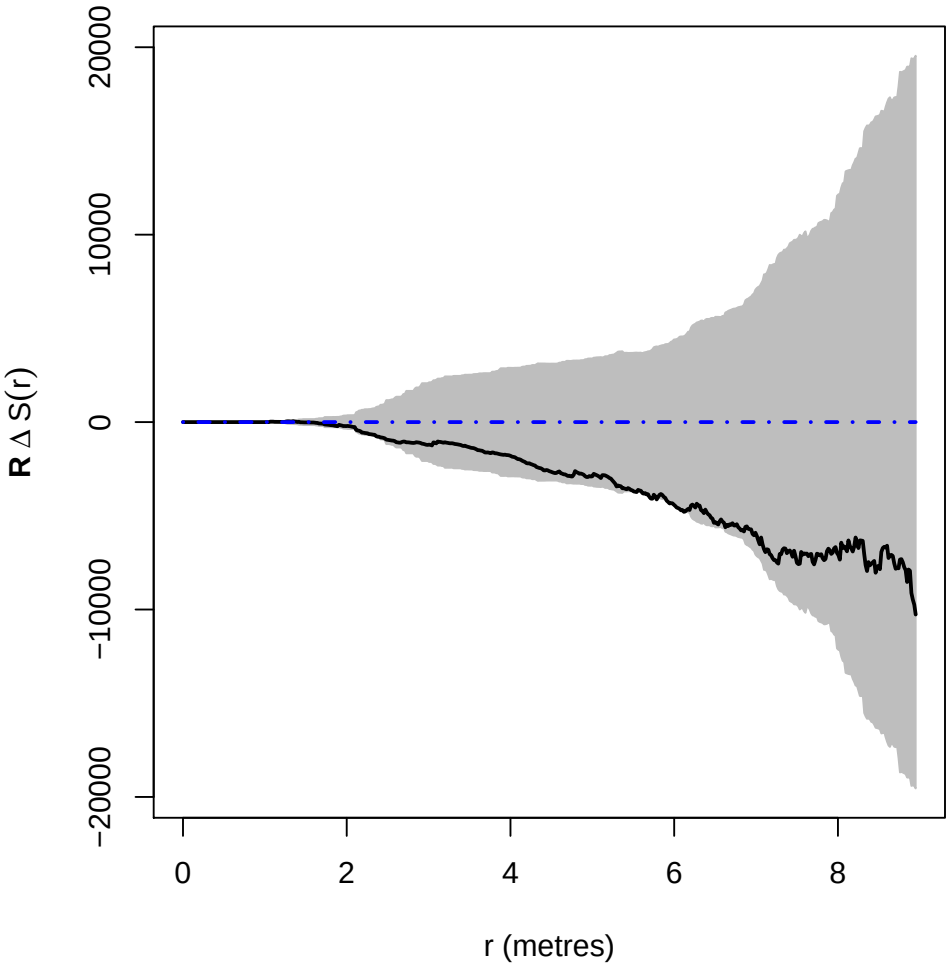
Here is the area pseudo-residual:

```
> Plot(a3, shade=Ashade)
```



Here is the triplets pseudo-residual:

```
> Plot(t3, shade=Tshade)
```



These plots suggest that the model has captured most of the interaction at short scales. At larger scales (say greater than 4 metres) there are additional inhibition effects which have not been captured by the model, but we shall not investigate them further.