Análisis de presencias con procesos de puntos

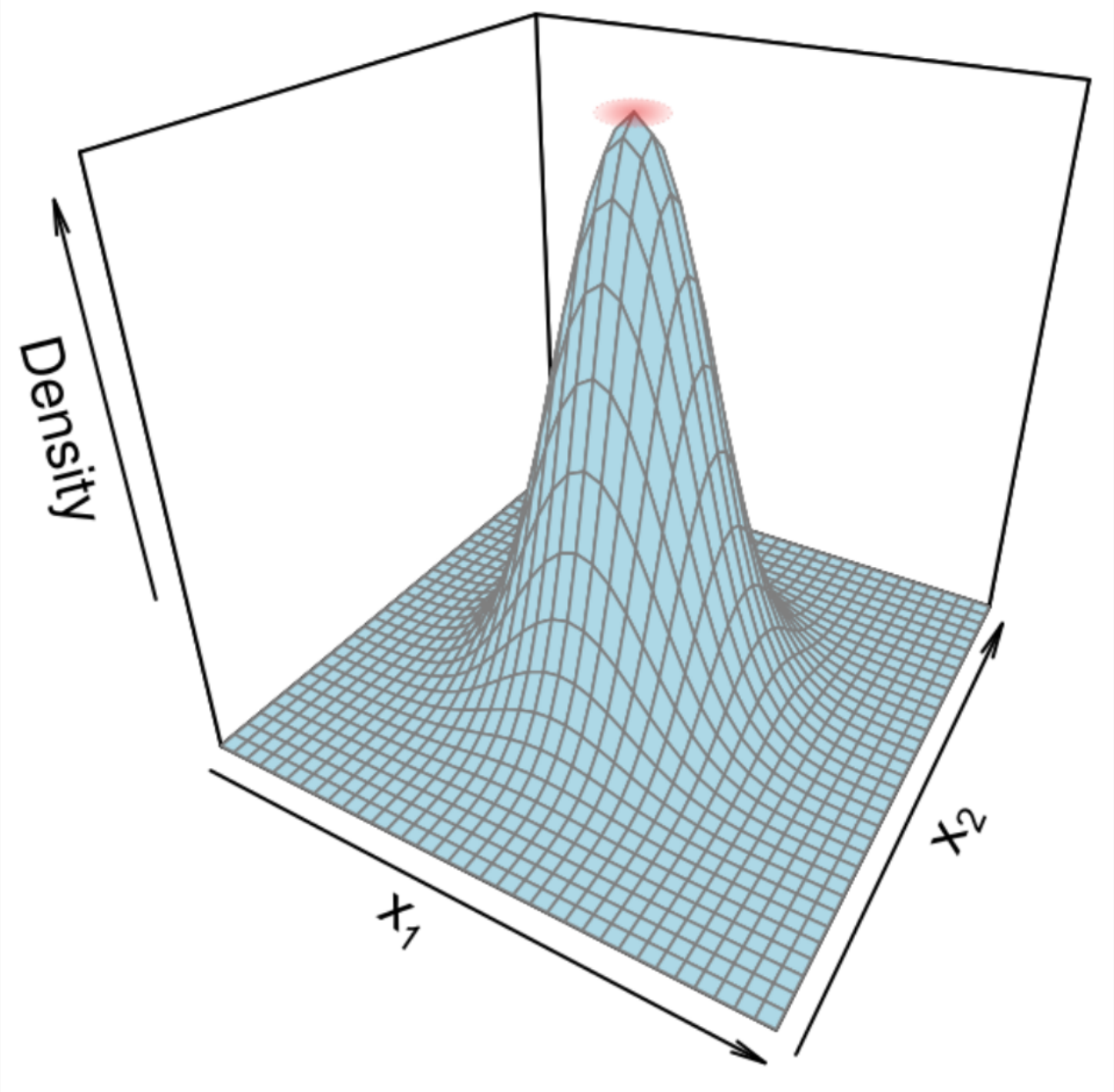
Tutorial intermedio de spatstat

Gerardo Martín

2022-06-29

Simulación de presencias

Especificación de un centroide



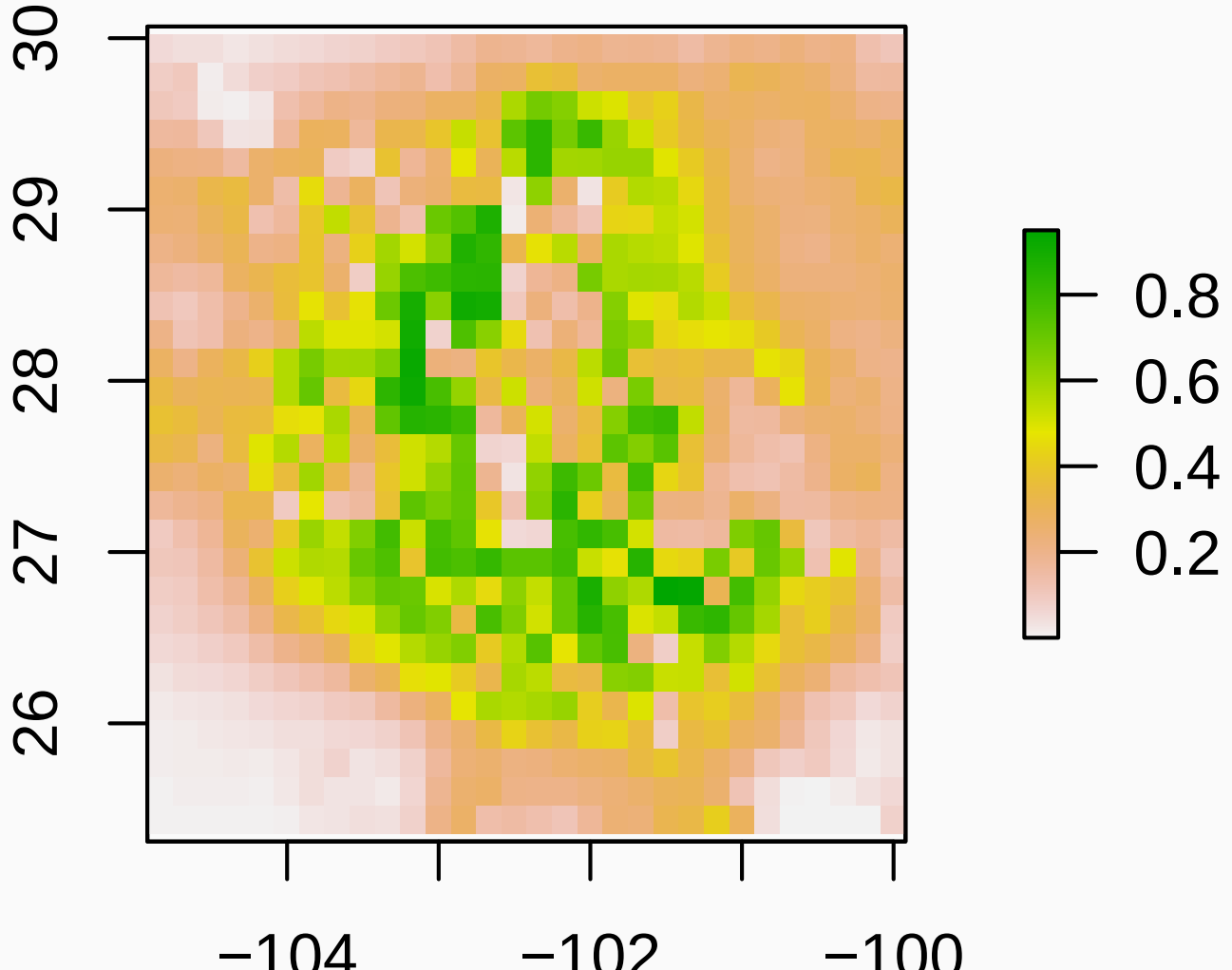
Código - generando favorabilidad “verdadera”

```
centroide <- cellStats(r, mean)
r.df <- data.frame(rasterToPoints(r))
covar <- cov(r.df[, 3:5])
md <- mahalanobis(r.df[, 3:5], center = centroide, cov = cova
head(md)
```

```
## [1] 5.846738 6.383437 6.443874 7.296541 6.475630 6.066614
```

Código - viendo la favorabilidad

```
md.r <- rasterFromXYZ(data.frame(r.df[, 1:2], md))
md.exp <- exp(-0.5*md.r)
plot(md.exp)
```

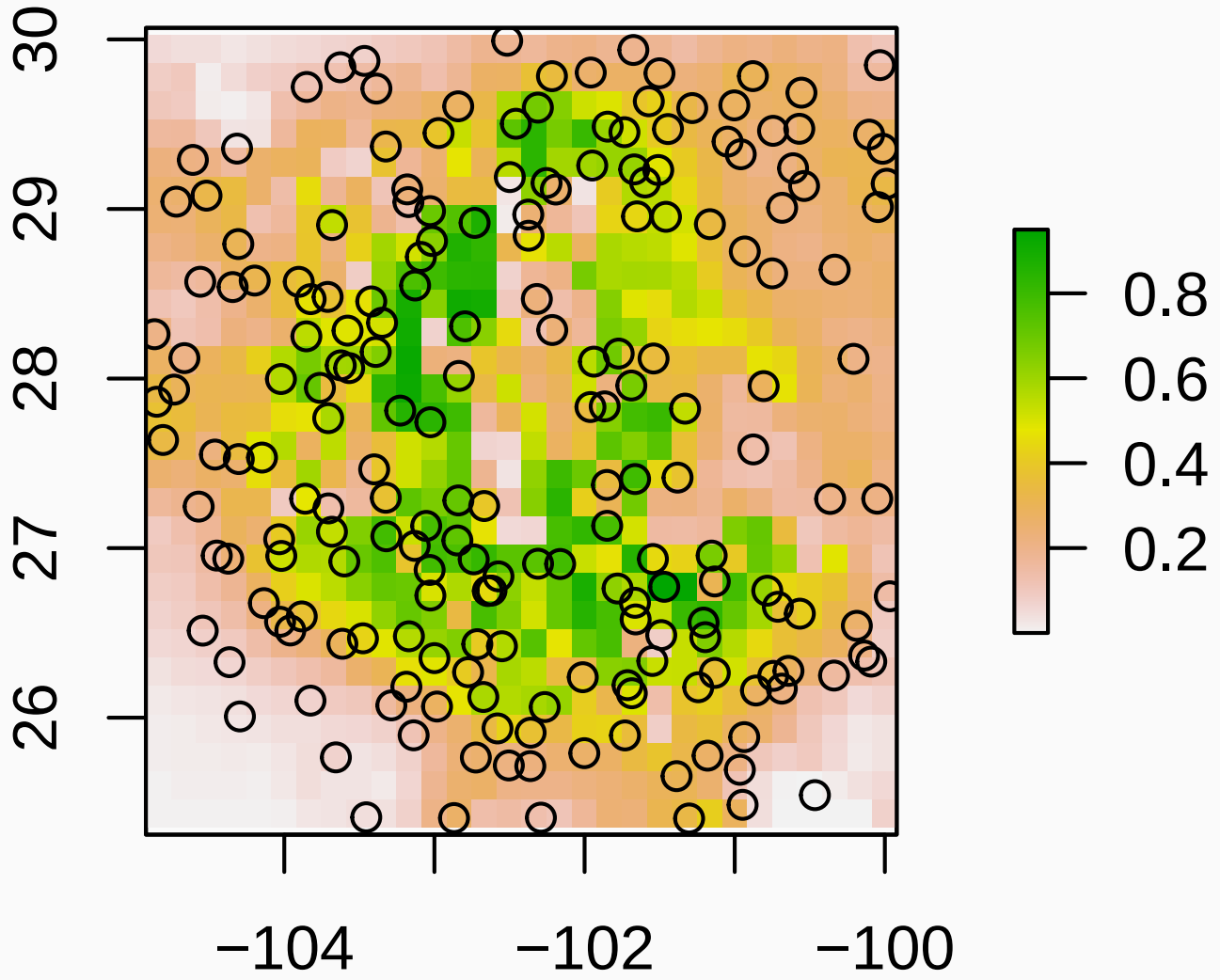


Código - simulando los puntos

```r
set.seed(182)
puntos.2 <- dismo::randomPoints(mask = md.exp,
                                n = 200,
                                prob = T)
puntos.2 <- data.frame(puntos.2)
puntos.2$x <- puntos.2$x + rnorm(200, 0, 0.05)
puntos.2$y <- puntos.2$y + rnorm(200, 0, 0.05)
```

Código - favorabilidad y puntos

```
plot(md.exp); points(puntos.2)
```



Formateo para spatstat

Cargando las funciones

```
source("Funciones-spatstat/imFromStack.R")
source("Funciones-spatstat/winFromRaster.R")
source("Funciones-spatstat/plotQuantIntens.R")
```

Formateo rápido

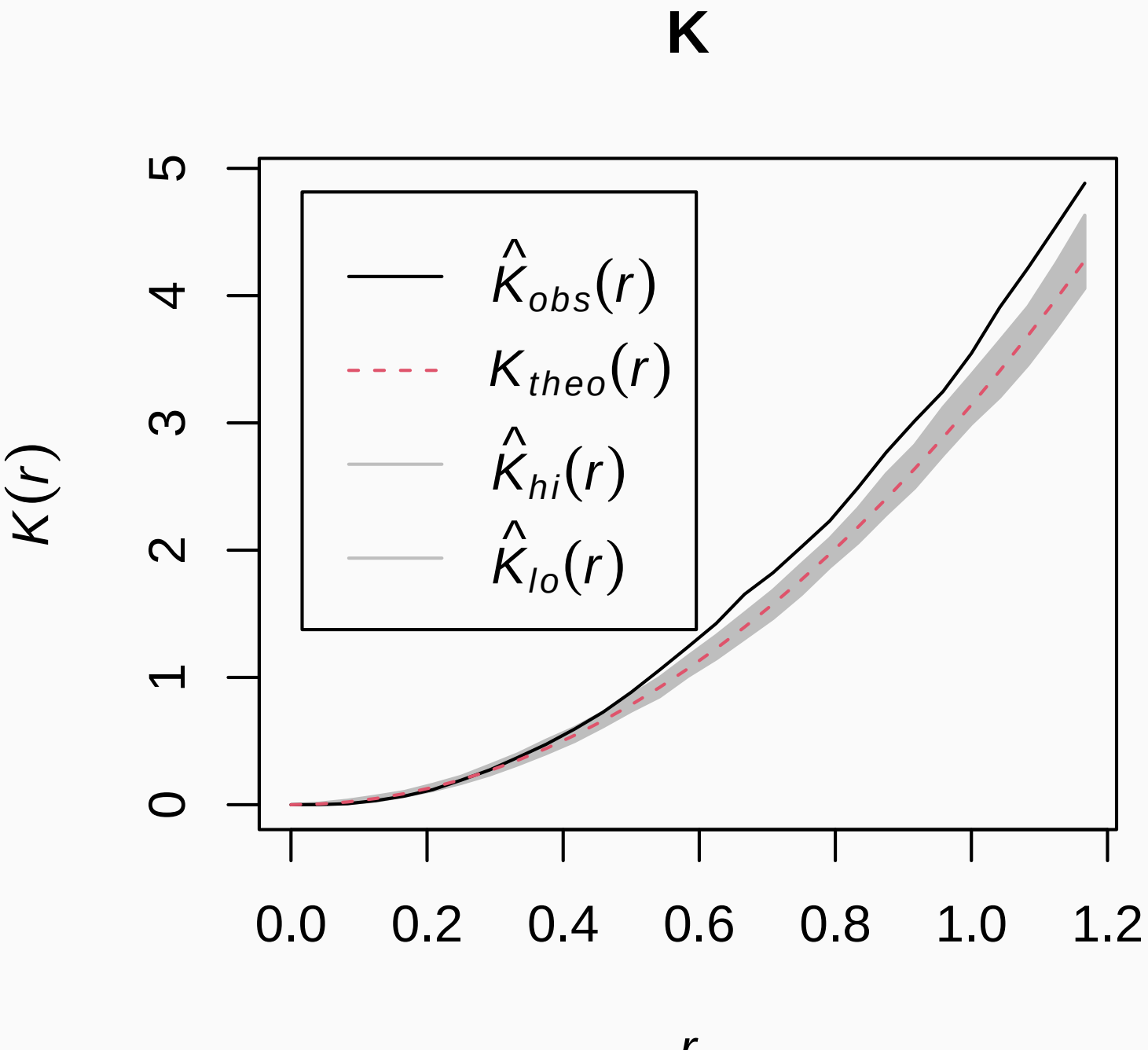
```
r.im <- imFromStack(r)
w <- winFromRaster(r)
puntos.2.ppp <- ppp(x = puntos.2$x,
                    y = puntos.2$y,
                    window = w,
                    check = F)
Q <- pixelquad(X = puntos.2.ppp, W = as.owin(w))
```

Análisis exploratorio

Autocorrelación

```
K <- envelope(puntos.2.ppp, fun = Kest, nsim = 39)
```

```
## Generating 39 simulations of CSR  ...
## 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 1
##
## Done.
```


K
$\hat{K}_{obs}(r)$
$K_{theo}(r)$
$\hat{K}_{hi}(r)$
$\hat{K}_{lo}(r)$
K(r)
0
1
2
3
4
5
0.0
0.2
0.4
0.6
0.8
1.0
1.2
r

Autocorrelación - notas

1. Pareciera que el proceso está levemente autocorrelacionado
2. No sabemos de momento si afectará al modelo
3. Debemos poner atención al modelo ajustado

Respuestas a variables

```
plotQuantIntens(imList = r.im,
                noCuts = 5,
                Quad = Q,
                p.pp = puntos.2.ppp,
                dir = "",
                name = "Respuestas-centroide")
```

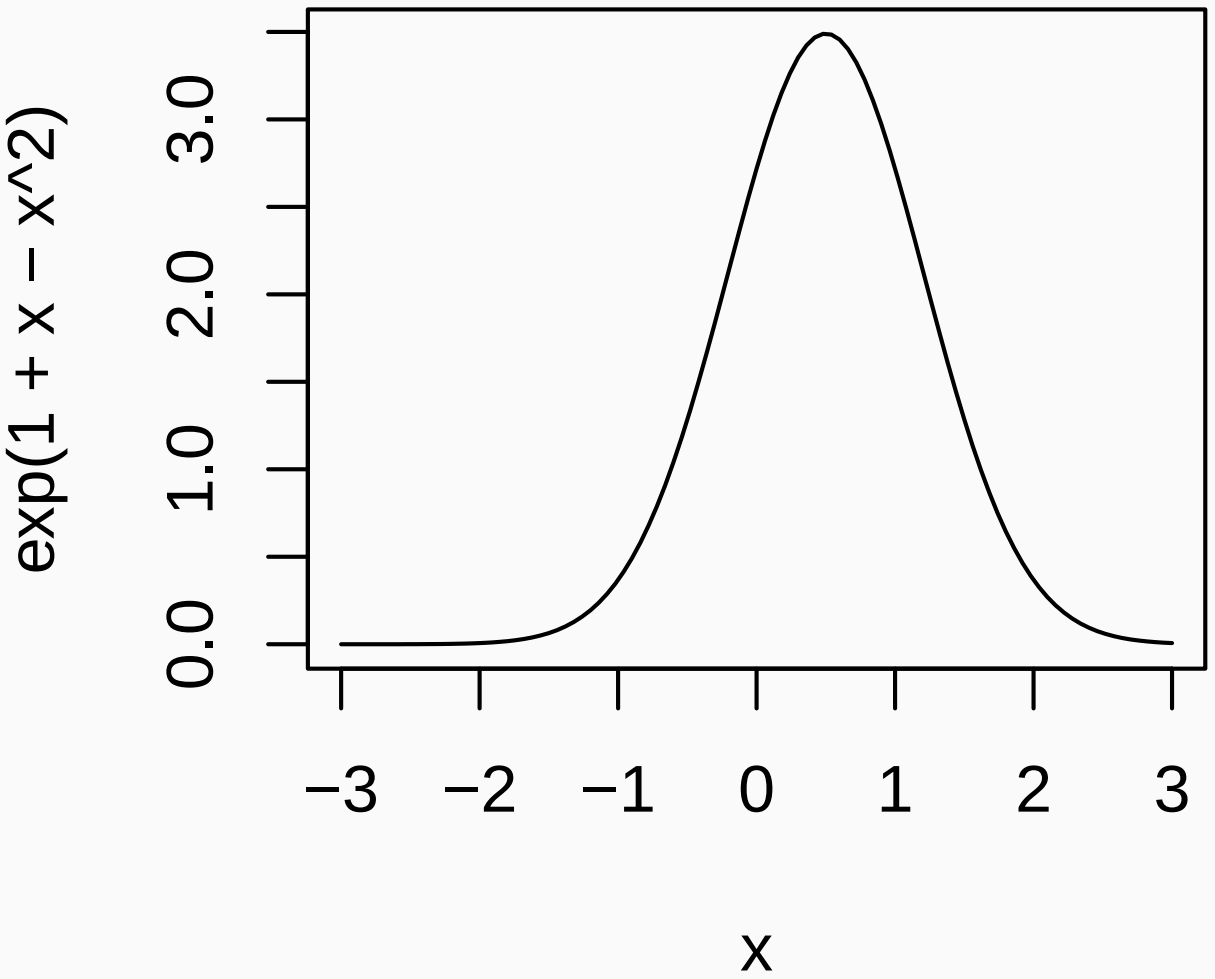
```
## pdf
##   2
```

Ver archivo de gráficas

Consideraciones para proponer modelos

Curvas con forma de campana $\longrightarrow$ fórmula cuadrática

```
curve(exp(1 + x - x^2), from = -3, 3)
```



Consideraciones para proponer modelos

Ecuación lineal:

$$y = \alpha + \beta_1 x_1 + \cdots + \beta_n x_n$$

Ecuación polinomial de 2^o grado

$$y = \alpha + \beta_1 x_1 + \beta_1' x_1^2 + \cdots + \beta_n x_n + \beta_n' x_n^2$$

Recordemos que $y = \log \lambda$

¿Qué variables podemos incluir en el mismo modelo?

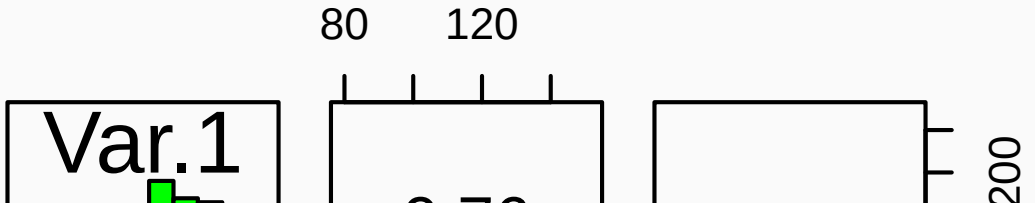
Regla de oro: Aquellas que no estén correlacionadas

- Que x_1 no sea predictor de x_2
- No se puede atribuir efecto de x_1 ó x_2 sobre λ
- Necesitamos medir correlación entre pares de variables (`pairs`)

Medición de correlación entre covariables

```
pairs(r)
```

```
## Warning in graphics::par(usr): argument 1 does not name a gra

## Warning in graphics::par(usr): argument 1 does not name a gra

## Warning in graphics::par(usr): argument 1 does not name a gra

## Warning in graphics::par(usr): argument 1 does not name a gra

## Warning in graphics::par(usr): argument 1 does not name a gra

## Warning in graphics::par(usr): argument 1 does not name a gra
```



Variables *compatibles*

Podemos incluir en el mismo modelo:

1. Var.1 y Var.3
2. Var.2 y Var.3

Por lo tanto las fórmula polinomial

$$\log \lambda = \alpha + \beta_1 x_1 + \beta_1' + x_1^2 + \beta_2 x_2 + \beta_2' + x_2^2 +$$

En **R**:

1. `~ Var.1 + Var.3 + I(Var.1^2) + I(Var.3^2)`
2. `~ Var.2 + Var.3 + I(Var.2^2) + I(Var.3^2)`

Ajustando los modelos

```
m1 <- ppm(Q = puntos.2.ppp,
          trend = ~ Var.1 + Var.3 + I(Var.1^2) + I(Var.3^2),
          covariates = r.im)
m2 <- ppm(Q = puntos.2.ppp,
          trend = ~ Var.2 + Var.3 + I(Var.2^2) + I(Var.3^2),
          covariates = r.im)
```

Comparando los modelos

```
AIC(m1); AIC(m2)
```

```
## [1] -473.2666
```

```
## [1] -468.7745
```

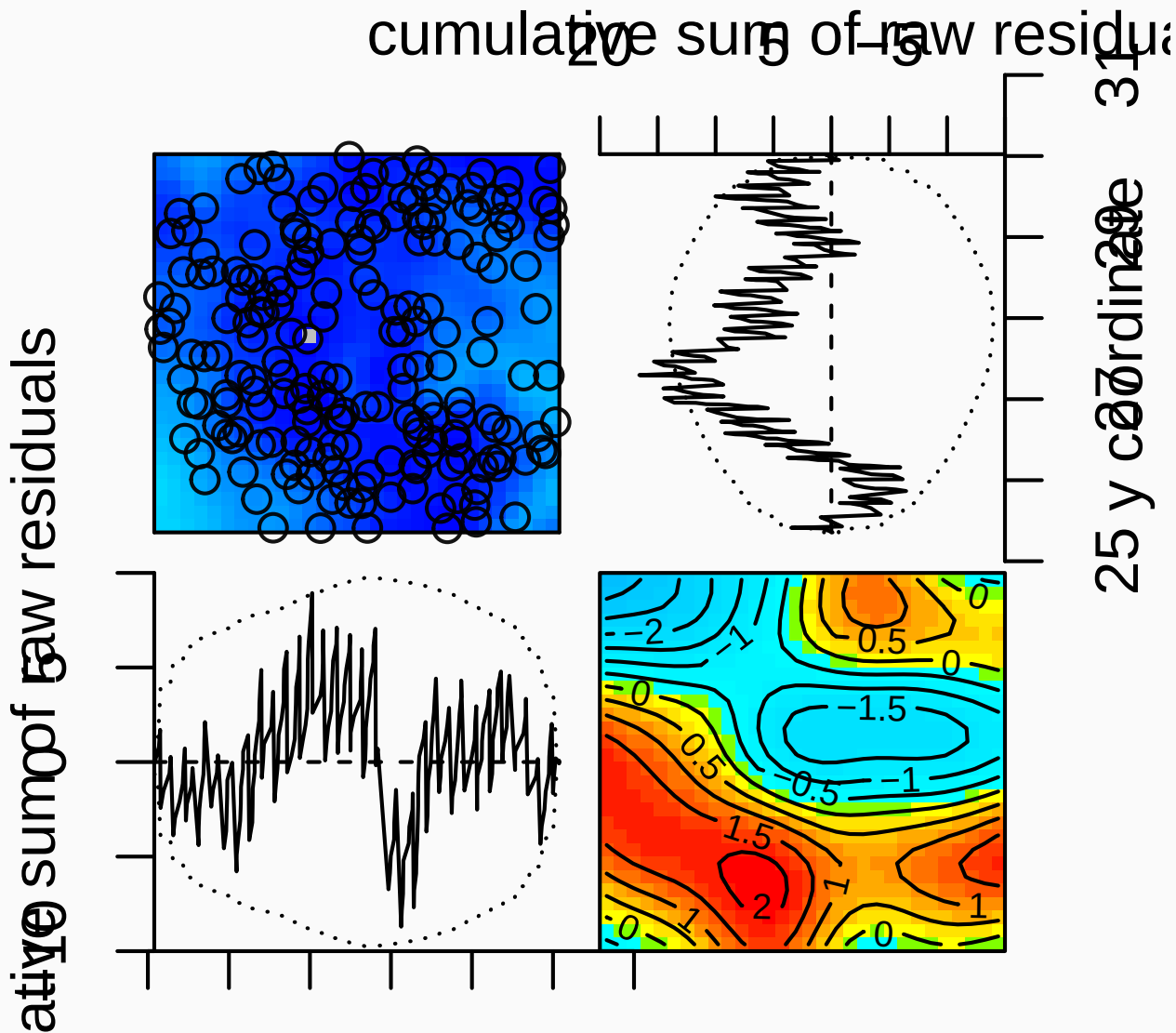
Analizar los efectos estimados

```
sum.m1 <- summary(m1)
knitr::kable(sum.m1$coefs.SE.CI[, 1:4])
```

	Estimate	S.E.	CI95.lo	CI95.hi
(Intercept)	-57.0868149	16.1815344	-88.8020396	-25.3715903
Var.1	0.4649718	0.1498312	0.1713081	0.7586355
Var.3	0.1973027	0.0970164	0.0071541	0.3874513
I(Var.1^2)	-0.0011954	0.0003795	-0.0019391	-0.0004517
I(Var.3^2)	-0.0006754	0.0003126	-0.0012881	-0.0000626

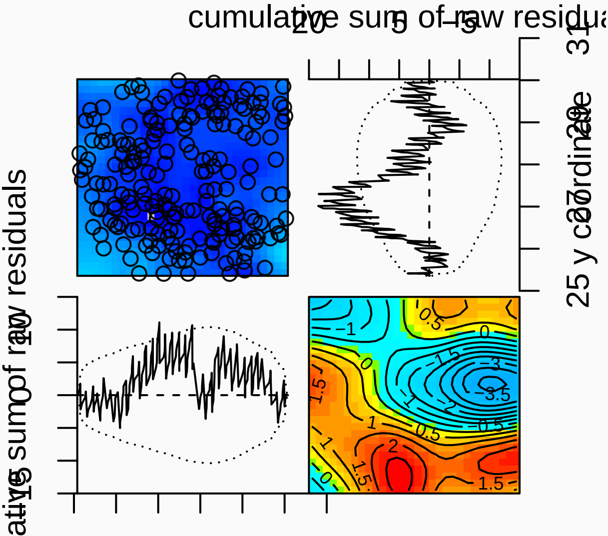
Diagnóstico - Residuales

```
par(mar = c(2,2,2,2))
diagnose.ppm(m1, main = "", cex.axis = 0.25)
```



Diangnóstico - Residuales

```r
par(mar = c(2,2,2,2))
diagnose.ppm(m2, main = "", cex.axis = 0.25)
```



Diagnóstico - Ripley

```
K1 <- envelope(m1, fun = Kest, nsim = 39)
```

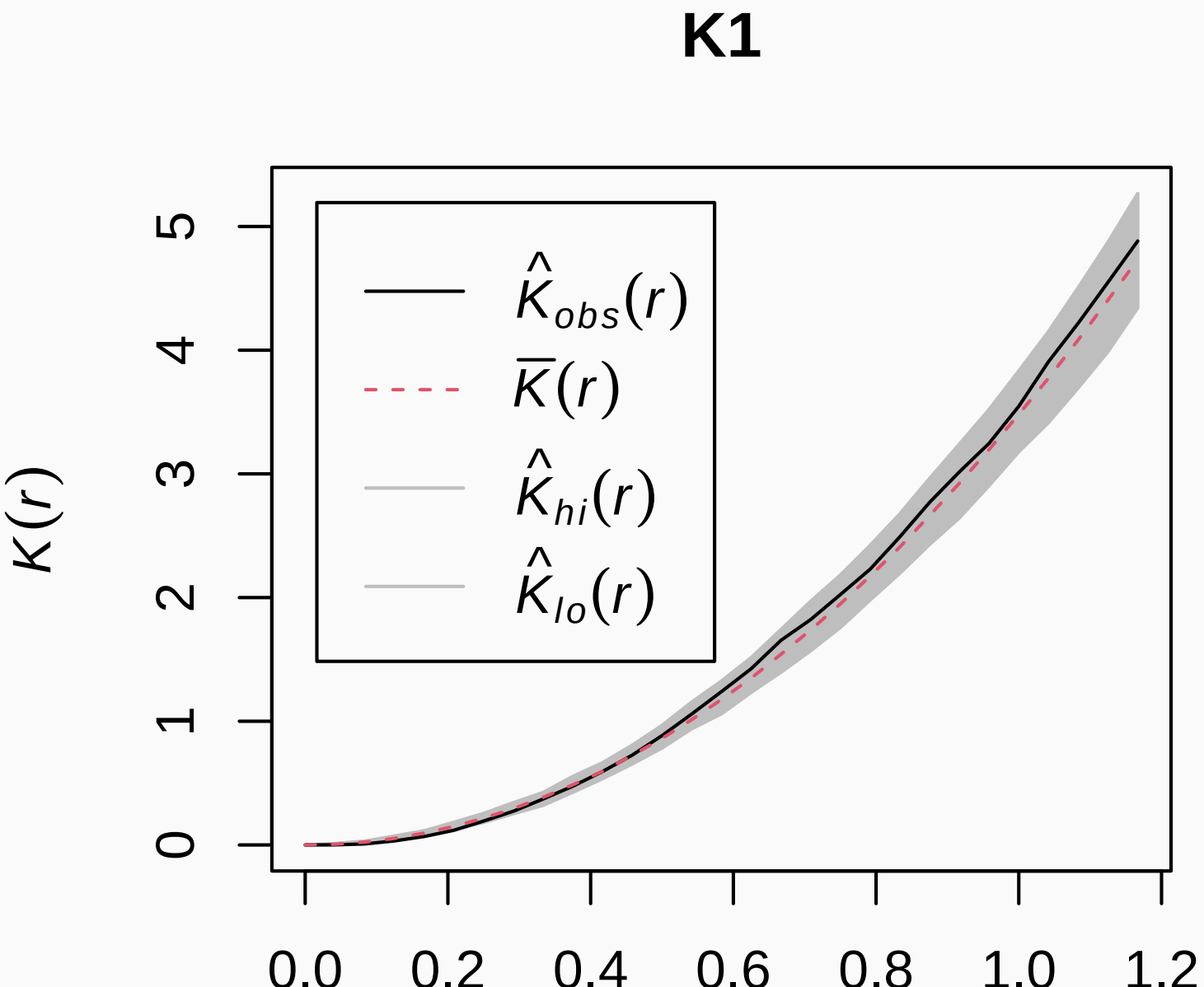
```
## Generating 39 simulated realisations of fitted Poisson model
## 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 1
##
## Done.
```

```
K2 <- envelope(m2, fun = Kest, nsim = 39)
```

```
## Generating 39 simulated realisations of fitted Poisson model
## 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 1
##
## Done.
```

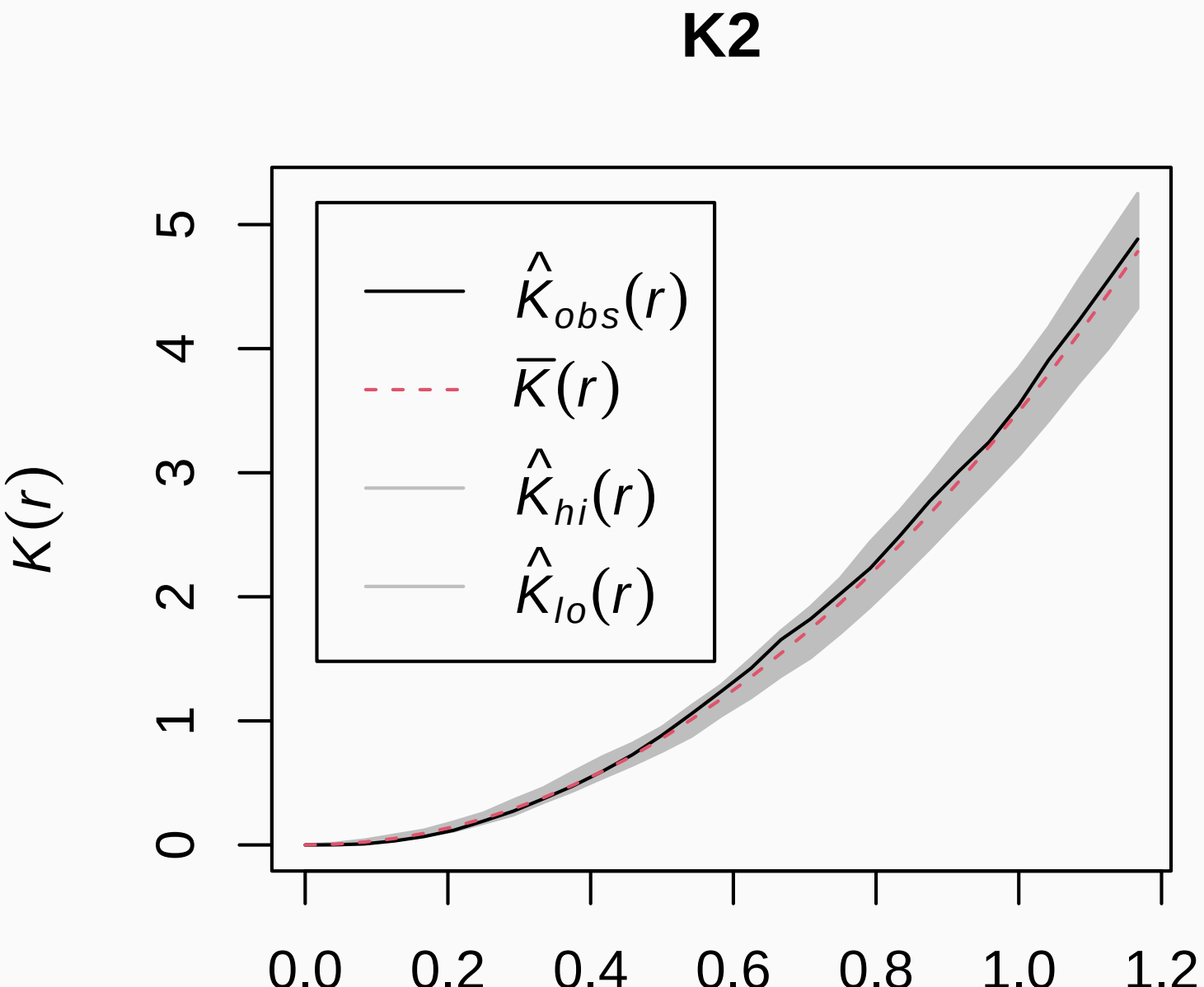
Diagnóstico - Ripley

```
plot(K1, cex = 0.5)
```



Diangóstico - Ripley

```
plot(K2, cex = 0.5)
```

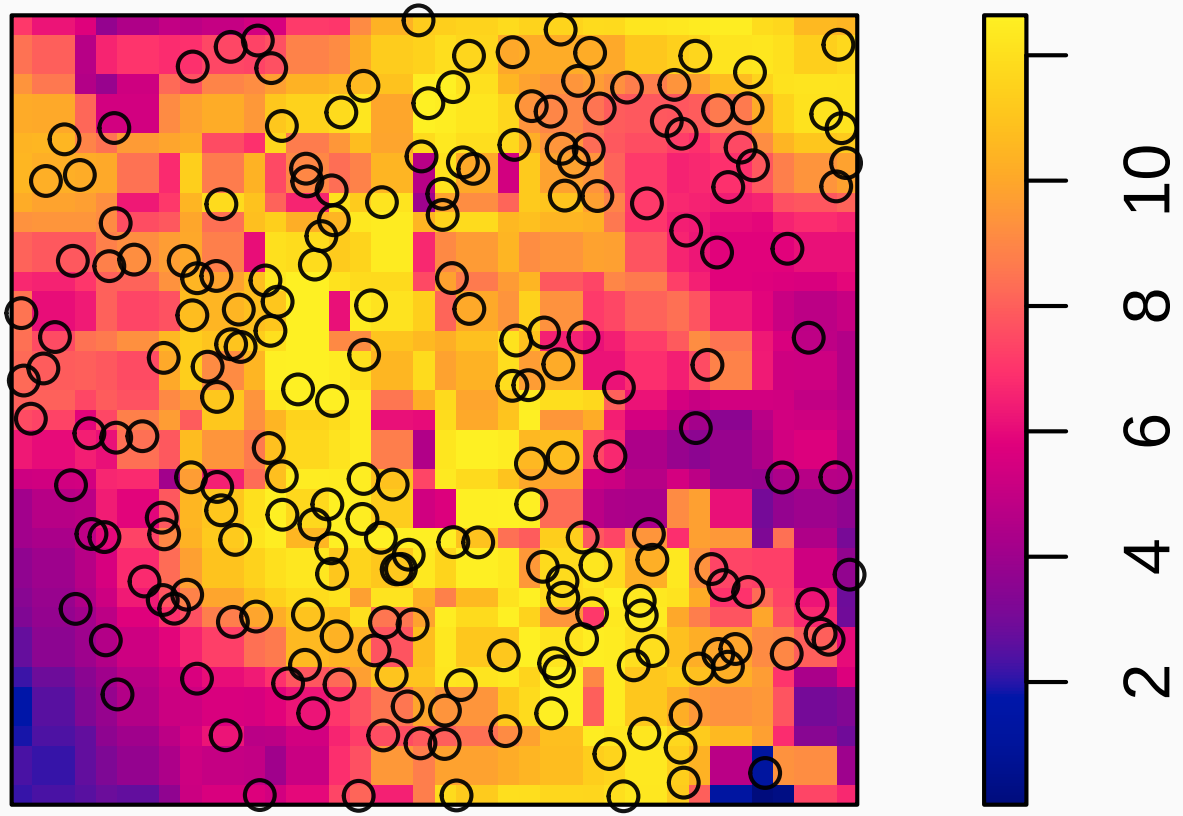


Resumen del análisis

- AIC menor para `m1`
- Residuales dentro de tolerancia para `m1`
- Prueba de ripley correcta para ambos modelos
    - No parece necesario modelar autocorrelación (lo haremos a continuación)
- Evidencia *favorece* a `m1`

Revisando la predicción

```
plot(m1, se = F, main = "")
```



Guardando los resultados

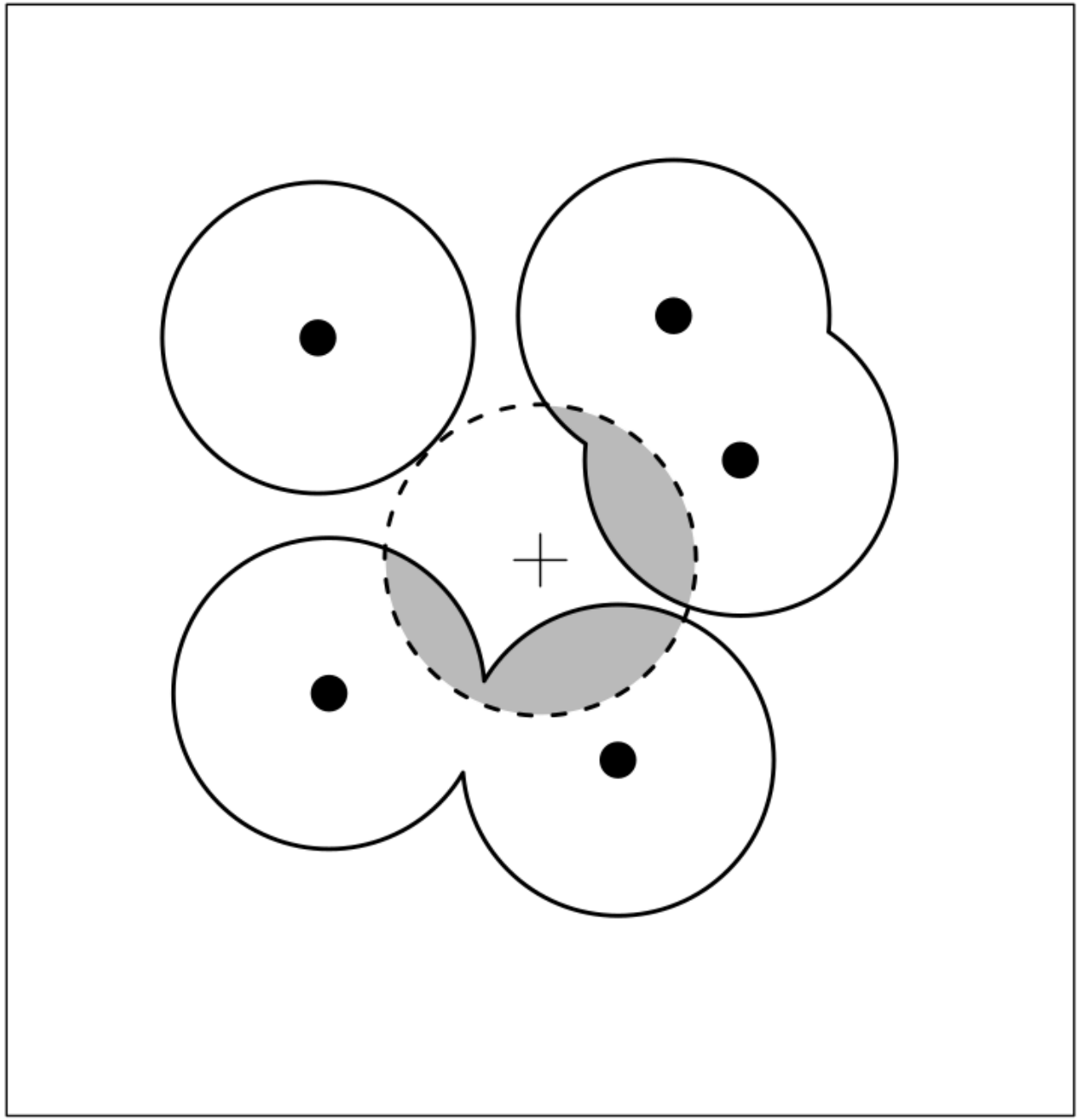
```
pred <- predict(m1)
pred.r <- raster(pred)
writeRaster(pred.r, "Predicción-m1", "GTiff",
            overwrite = T)
```

Modelando los efectos espaciales

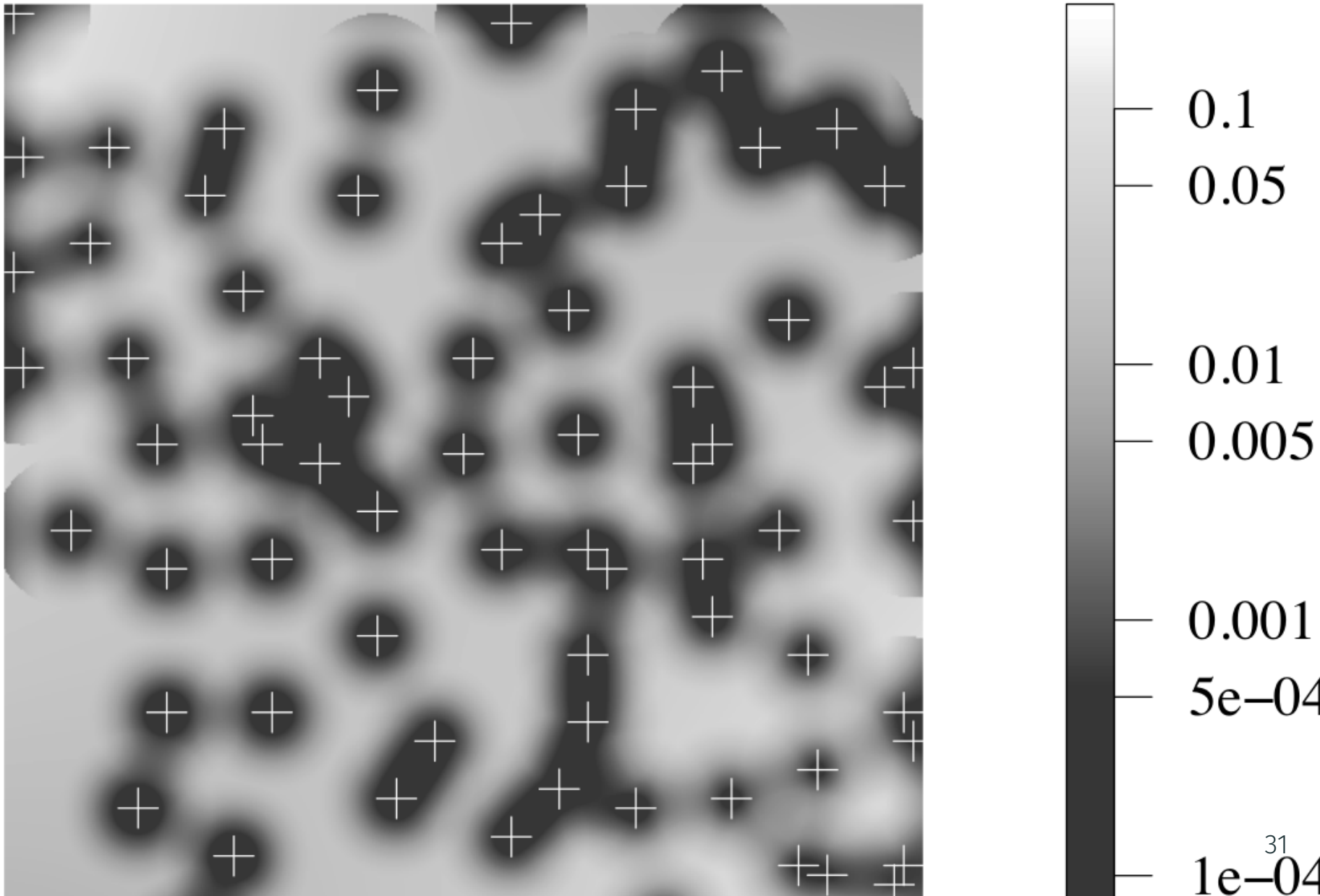
Modelos de interacción

- Estiman efecto aleatorio para puntos cercanos
- Sirven para procesos de exclusión o agregación moderada
- Hay varios tipos de interacciones entre puntos

¿Qué es interacción?



Tipos de interacciones



FUNCTION	MODEL
AreaInter	area-interaction process
BadGey	multiscale Geyer saturation proce
Concom	connected component interaction
Geyer	Geyer saturation process
Hybrid	hybrid of several interactions
Ord	Ord model, user-supplied potentia
OrdThresh	Ord model, threshold potential
Saturated	saturated model, user-supplied po
SatPiece	multiscale saturation process
Triplets	Geyer triplet interaction process

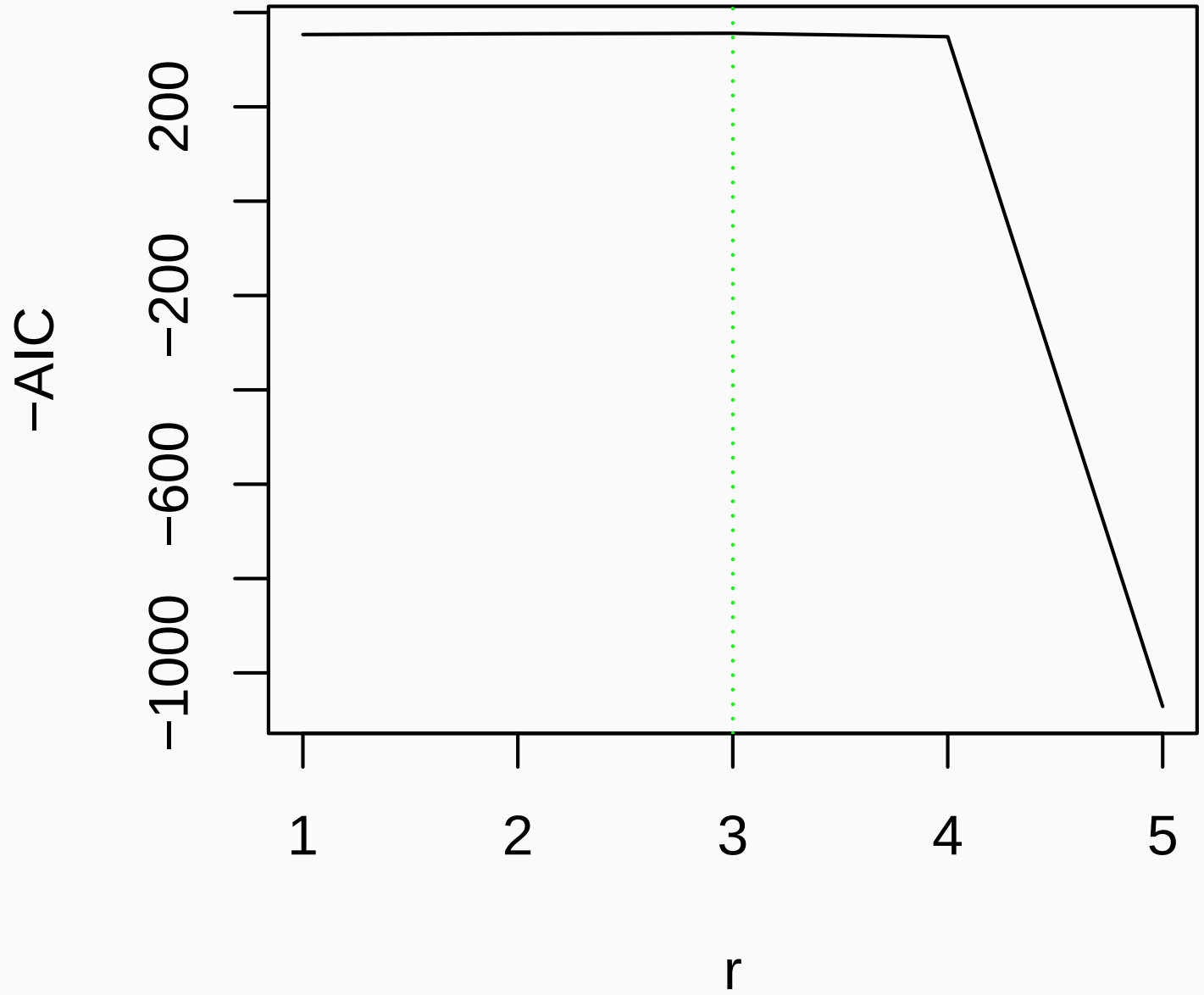
Para generar un modelo de interacción

1. Establecer tamaño del búfer

```
rr <- data.frame(r=seq(1,5,by=1))
p <- profilepl(rr, Strauss,
               puntos.2.ppp ~ Var.1 + Var.3 + I(Var.1^2) + I(
          covariates = r.im, aic=T, rbord = 0.5)

## comparing 5 models...

## 1, 2, 3, 4,  5.

## fitting optimal model...

## done.
```

Para generar un modelo de interacción

```
plot(p, main = "")
```



Para generar un modelo de interacción

Un radio de tamaño 2 minimiza la pseudo-verosimilitud, de modo que el modelo de interacción con la fórmula de m1 es:

```
m1.int <- ppm(Q = puntos.2.ppp,
              trend = ~ Var.2 + Var.3 + I(Var.2^2) + I(Var.3^2),
              covariates = r.im,
              Strauss(p$iopt), rbord = 1) #Interacción
```

Efectos estimados

```
sum.int <- summary(m1.int)
knitr::kable(sum.int$coefs.SE.CI[, 1:4])
```

	Estimate	S.E.	CI95.lo	CI95.hi
(Intercept)	-68.4382995	38.7998381	-144.4845847	7.6079858
Var.2	0.5668834	0.2965100	-0.0142655	1.1480322
Var.3	0.5331550	0.4226373	-0.2951989	1.3615089
I(Var.2^2)	-0.0024765	0.0012598	-0.0049457	-0.0000074
I(Var.3^2)	-0.0017398	0.0014341	-0.0045507	0.0010710
Interaction	-0.0127375	0.0102314	-0.0327907	0.0073157

Efectos estimados - comparación

```
coef(m1)
```

```
##  (Intercept)       Var.1       Var.3   I(Var.1^2)   I(Var.3^2
## -5.708681e+01  4.649718e-01  1.973027e-01 -1.195420e-
03 -6.753759e-04
```

```
coef(m1.int)
```

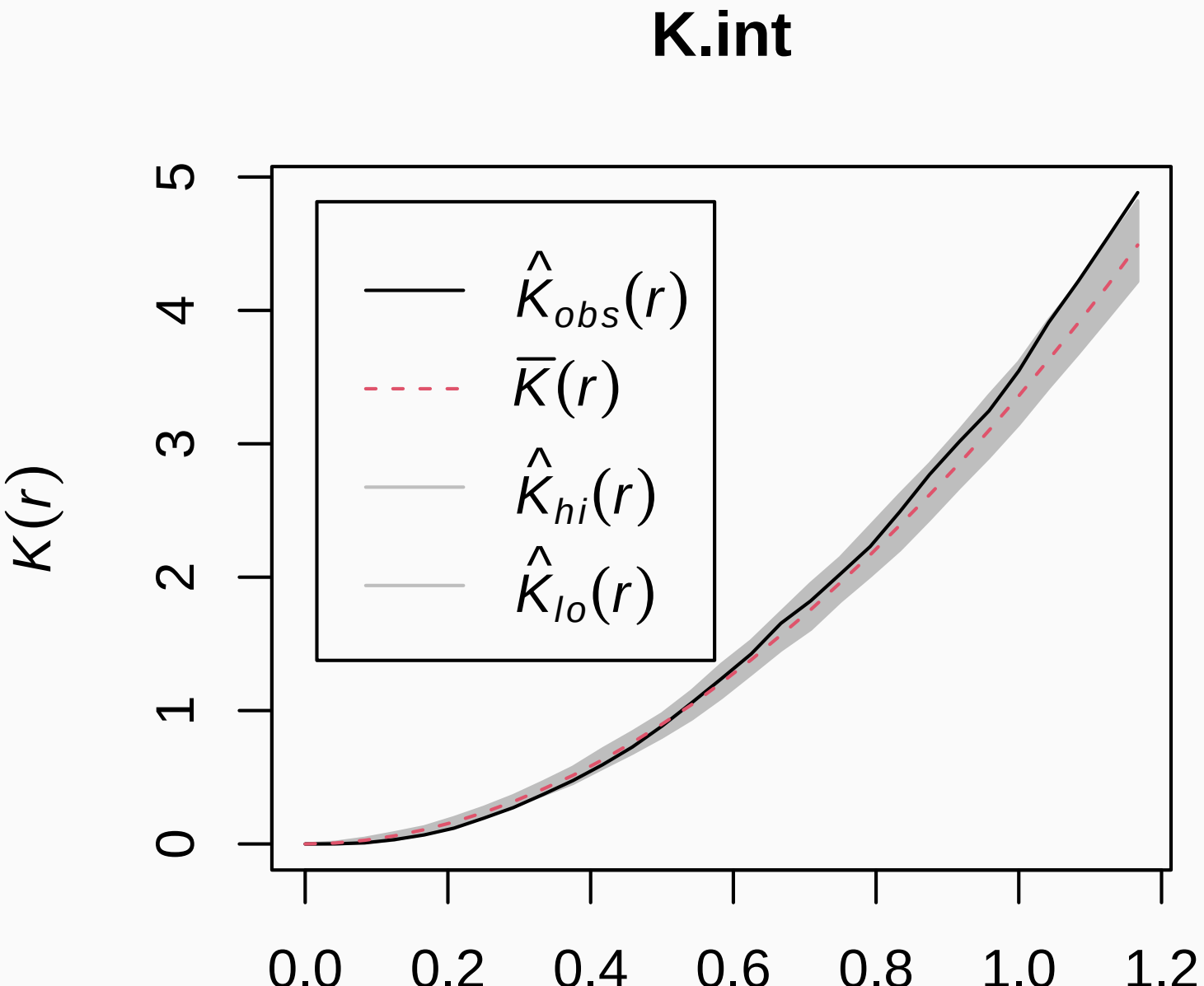
```
##  (Intercept)       Var.2       Var.3   I(Var.2^2)   I(Var.3^2
##  -68.438299459       0.566883377       0.533154977    -
0.002476525  -0.001739840
##   Interaction
##  -0.012737533
```

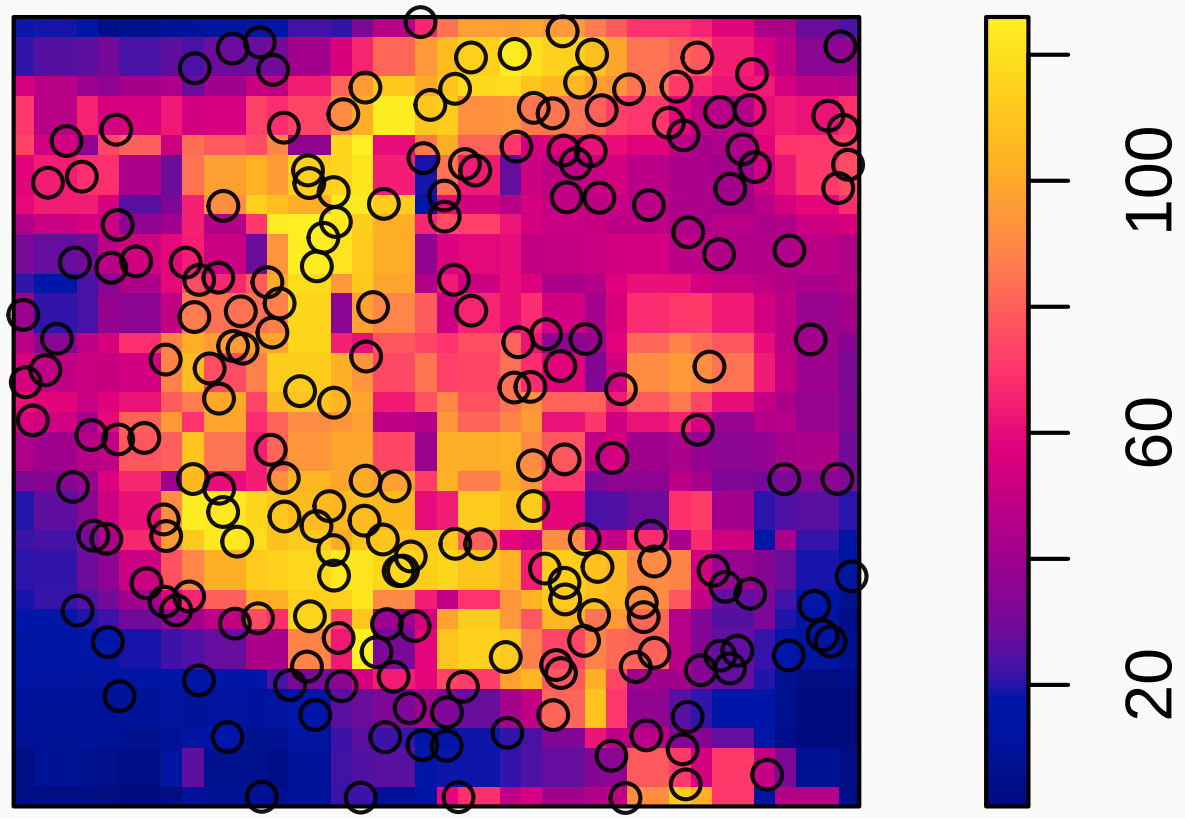
Diangóstico

```
K.int <- envelope(m1.int, Kest, nsim = 39)
```

```
## Generating 39 simulated realisations of fitted Gibbs model .
## 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 1
##
## Done.
```

```
plot(K.int)
```


K.int
K(r)
0
1
2
3
4
5
0.0
0.2
0.4
0.6
0.8
1.0
1.2
$\hat{K}_{obs}(r)$
$\overline{K}(r)$
$\hat{K}_{hi}(r)$
$\hat{K}_{lo}(r)$


Fitted trend
100
60
20

Proceso Cox log-Gaussiano

¿Qué es?

- En MPPs
    - Intensidad es explicada por covariables si
    - Covariables rara vez explican puntos agregados
- Gaussiano = Distribución normal
    - Efecto aleatorio con distribución normal multivariada

¿Qué es?

$$\log \lambda_i = \alpha + \beta_1 x_{1,i} + \cdots + G(u_i, v_i)$$

- α es el intercepto global - $G(u_i)$ es el intercepto aleatorio para cada píxel - Cuando todas las $x = 0$, la intensidad en el píxel i es $\exp(\alpha + G(u_i))$

¿Con qué se ajusta un LGCP en R?

- Frecuentista - `spatstat` (rápido poco preciso)
- Bayesiano
    - `RINLA` (moderadamente rápido, moderadamente preciso)
    - `lgcp` (muuuuy lento, bastante preciso)
- Frecuentista son aproximaciones, y Bayesiano son estimaciones *verdaderas*

Ajustando un LGCP con spatstat

```
m1.lgcp <- kppm(puntos.2.ppp,
                trend = ~ Var.2 + Var.3 + I(Var.2^2) + I(Var.
                covariates = r.im,
                clusters = "Thomas",
                statistic = "K", # K de Ripley
                method = "clik2", # Contraste con K
                model = "exp") # Modelo de varianza
```

Ajustando un LGCP con `spatstat`

```
sum.lgcp <- summary(m1.lgcp)
knitr::kable(sum.lgcp$coefs.SE.CI[, 1:4])
```

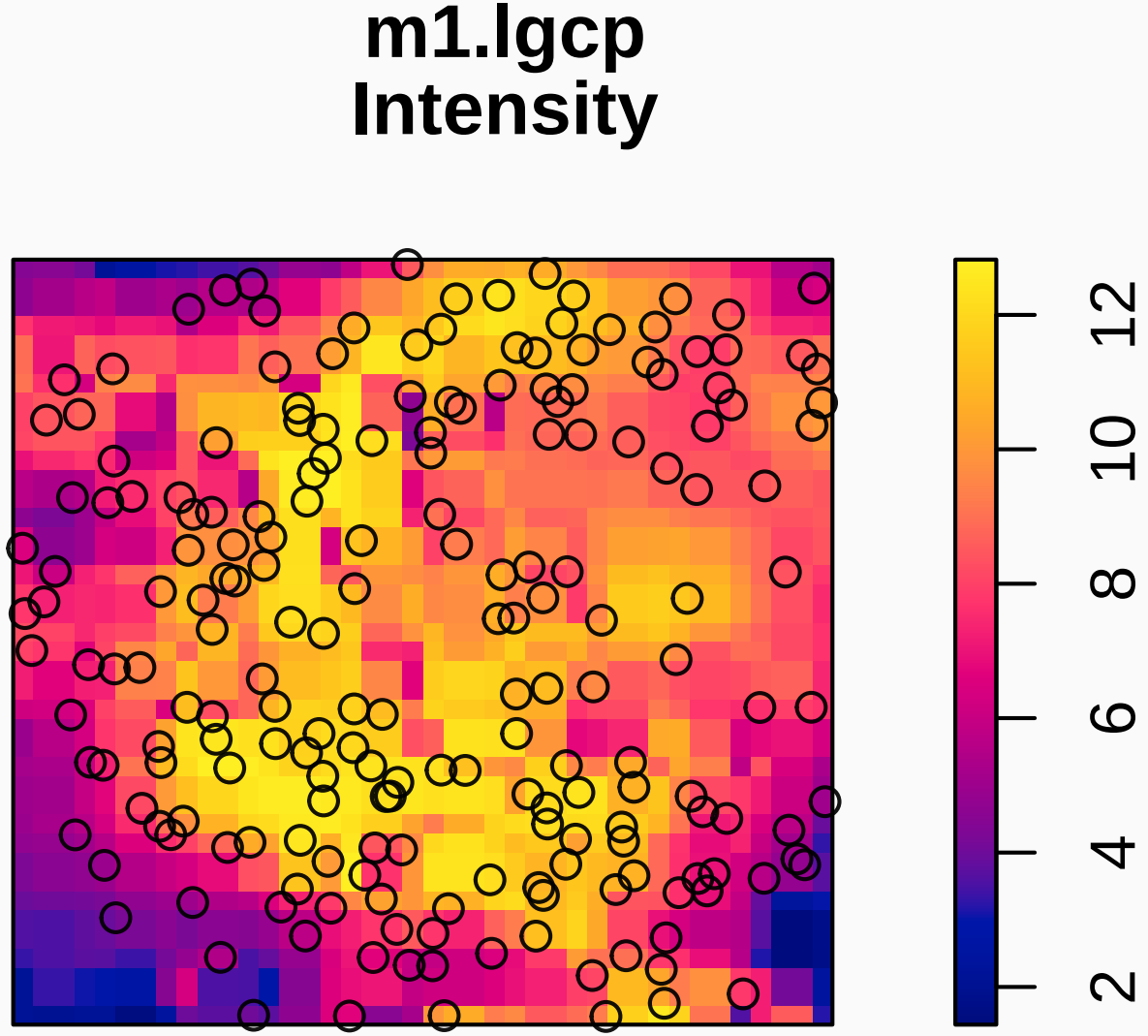
	Estimate	S.E.	CI95.lo	CI95.hi
(Intercept)	-32.6841705	10.0403861	-52.3629657	-13.0053753
Var.2	0.2686269	0.0933978	0.0855706	0.4516833
Var.3	0.2552085	0.0994261	0.0603370	0.4500800
I(Var.2^2)	-0.0011466	0.0003974	-0.0019254	-0.0003678
I(Var.3^2)	-0.0008349	0.0003187	-0.0014595	-0.0002102

Comparando con MPP

```
knitr::kable(sum.m1$coefs.SE.CI[, c(1, 2, 3, 4)])
```

	Estimate	S.E.	CI95.lo	CI95.hi
(Intercept)	-57.0868149	16.1815344	-88.8020396	-25.3715903
Var.1	0.4649718	0.1498312	0.1713081	0.7586355
Var.3	0.1973027	0.0970164	0.0071541	0.3874513
I(Var.1^2)	-0.0011954	0.0003795	-0.0019391	-0.0004517
I(Var.3^2)	-0.0006754	0.0003126	-0.0012881	-0.0000626

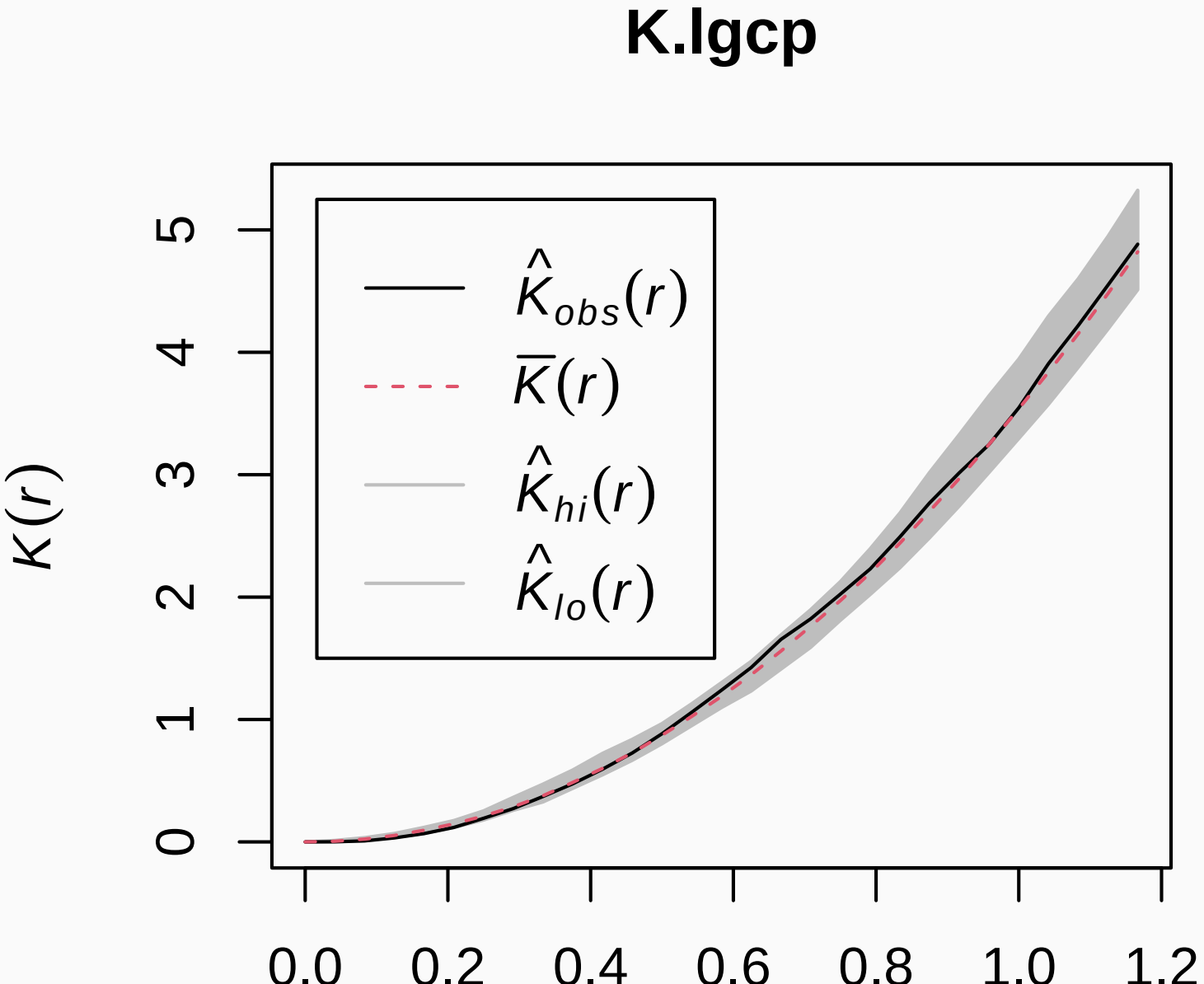
```
plot(m1.lgcp, what = "intensity")
```



Diagnóstico

```
K.lgcp <- envelope(m1.lgcp, Kest, nsim = 39)

## Generating 39 simulated realisations of fitted cluster model
## 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 1
##
## Done.
```

```
plot(K.lgcp)
```


K.lgcp
K(r)
0
1
2
3
4
5
0.0
0.2
0.4
0.6
0.8
1.0
1.2
$\hat{K}_{obs}(r)$
$\overline{K}(r)$
$\hat{K}_{hi}(r)$
$\hat{K}_{lo}(r)$

Conclusiones

- Modelo Poisson
    - Más simple, y no parece tener problemas
    - IC de estimaciones más amplios que LGCP
- Interacción
    - IC más amplios que MPP
- LGCP
    - Función K más cercana a expectativa teórica ### Alternativas de modelación
- Respuestas bisagra: Regresión por partes
- Respuestas no lineales: Suavizadores GAM
- Interacciones entre variables