Project

Tianyi Xia, Zetian Zhao, Litao Zheng

22 April, 2025 10:36:03

Species => `Great Blue Heron`.

Spatial Analysis

```
# Load libraries
library(spatstat)
```

```
## Warning: package 'spatstat' was built under R version 4.4.3

## Loading required package: spatstat.data

## Warning: package 'spatstat.data' was built under R version 4.4.3

## Loading required package: spatstat.univar

## Warning: package 'spatstat.univar' was built under R version 4.4.3

## spatstat.univar 3.1-2

## Loading required package: spatstat.geom

## Warning: package 'spatstat.geom' was built under R version 4.4.3

## spatstat.geom 3.3-6

## Loading required package: spatstat.random

## Warning: package 'spatstat.random' was built under R version 4.4.3

## spatstat.random 3.3-3

## Loading required package: spatstat.explore

## Warning: package 'spatstat.explore' was built under R version 4.4.3

## Loading required package: nlme
```

```
## spatstat.explore 3.4-2

## Loading required package: spatstat.model

## Warning: package 'spatstat.model' was built under R version 4.4.3

## Loading required package: rpart

## spatstat.model 3.3-5

## Loading required package: spatstat.linnet

## Warning: package 'spatstat.linnet' was built under R version 4.4.3

## spatstat.linnet 3.2-5

##
## spatstat 3.3-2
## For an introduction to spatstat, type 'beginner'
```

```
library(sf)
```

```
## Linking to GEOS 3.12.1, GDAL 3.8.4, PROJ 9.3.1; sf_use_s2() is TRUE
```

```
library(raster)
```

```
## Warning: package 'raster' was built under R version 4.4.3

## Loading required package: sp

## Warning: package 'sp' was built under R version 4.4.3

##
## Attaching package: 'raster'

## The following object is masked from 'package:nlme':
##
##     getData
```

Load the Covariates

```
load("BC_Covariates.Rda")
ls()
```

```
## [1] "DATA"
```

Summary

```r
summary(DATA)
```

```
##             Length Class           Mode
## Window       1     SpatialPolygons S4
## Elevation   10     im              list
## Forest      10     im              list
## HFI         10     im              list
## Dist_Water  10     im              list
```

Pre-process

```r
# pre-process the csv file
species_data <- read.delim("species.csv", sep = "\t", header = TRUE)
species_data <- species_data[, colSums(is.na(species_data)) < nrow(species_data)]
```

```r
# Select the key columns for spatial analysis
species_subset <- species_data[, c("scientificName", "decimalLongitude", "decimalLatitude", "coordinateU

# Filter out records with missing coordinates
species_clean <- species_subset[!is.na(species_subset$decimalLongitude) & !is.na(species_subset$decimall
```

Begin Analysis

```r
window <- DATA$Window
class(DATA$Window)
```

```
## [1] "SpatialPolygons"
## attr(,"package")
## [1] "sp"
```

```r
window_sf <- st_as_sf(window)
window_owin <- as.owin(window_sf)
```

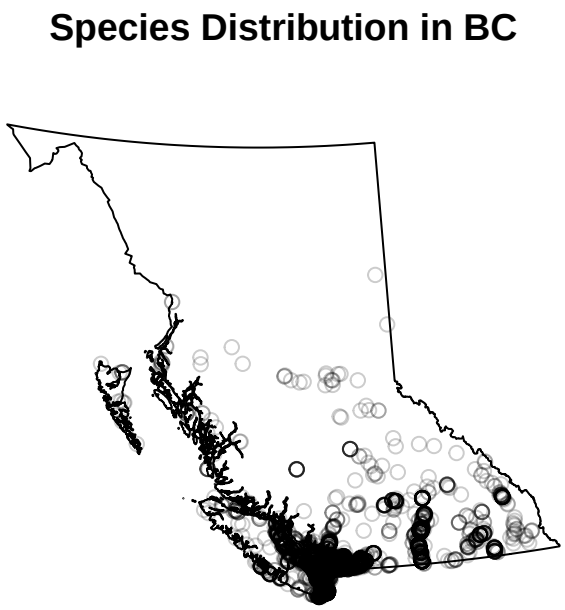
PPP-Object

```r
# Convert your species coordinates to match the BC window projection
library(sp)

# Create a SpatialPoints object with your species data
species_sp <- SpatialPoints(
  coords = data.frame(x = species_clean$decimalLongitude, y = species_clean$decimalLatitude),
  proj4string = CRS("+proj=longlat +datum=WGS84")  # GBIF typically uses WGS84
)

# Transform to match the projection of your BC window
```

```r
species_transformed <- spTransform(species_sp, CRS(proj4string(window)))

# Extract the transformed coordinates
transformed_coords <- coordinates(species_transformed)

# Create the ppp object with transformed coordinates
x_coordinates <- transformed_coords[,1]
y_coordinates <- transformed_coords[,2]

# make a df for the species
species_df <- data.frame(x = x_coordinates, y = y_coordinates)

# Check which points are inside the owin window
inside <- inside.owin(x_coordinates, y_coordinates, window_owin)

# filter the points out that match the window
species_df <- species_df[inside, ]

# create the ppp-object
species_ppp <- ppp(
  x = species_df$x,
  y = species_df$y,
  window = window_owin
)
```

```
## Warning: data contain duplicated points
```

```r
# plot the window with points
plot(species_ppp, main = "Species Distribution in BC")
```



Conclusion:

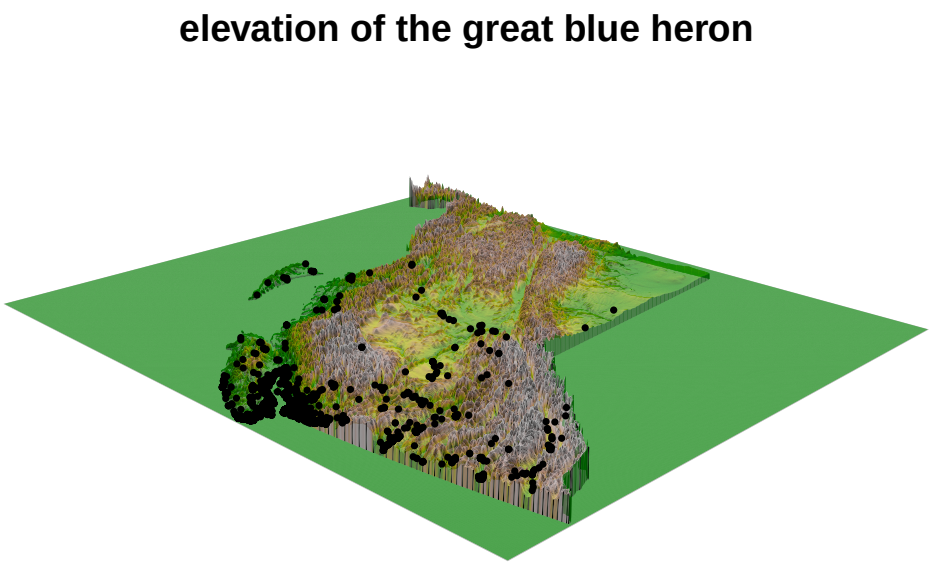
First impression => the distribution is not homo.

3D Map with Elevation

```
# get the elevation from DATA
elevation <- DATA$Elevation
figure_3d <- persp(elevation,
      # horizontal and vertical rotation.
      theta = 45,
      phi = 20,
      # z-axis expansion (need to be bigger for the BC)
      expand = 25,
      # remove border
      border = NA,
      # add a base
      apron = TRUE,
      # shade intensity
      shade = 0.4,
      # axes off.
      box = FALSE,
      main = "elevation of the great blue heron",
      # allow the overlaying for the parks location.
      visible = TRUE,
```

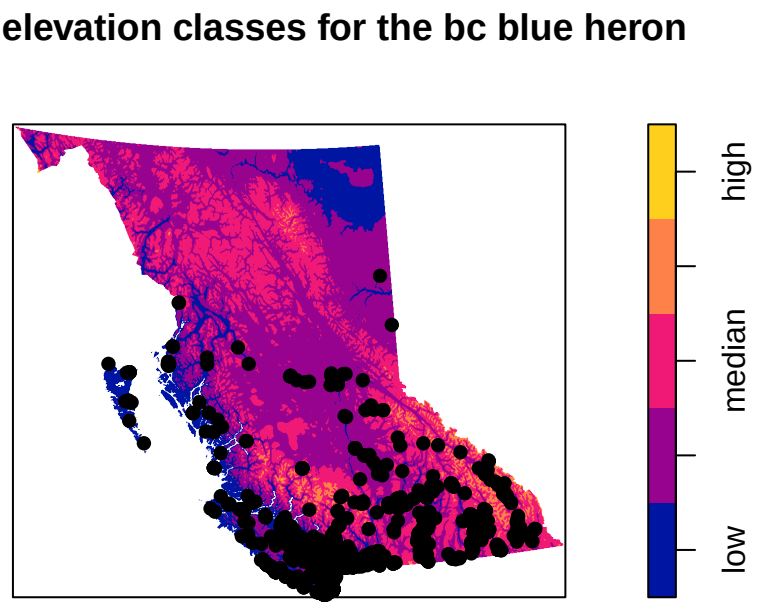
```
      # use the terrain color pallete which is built-in for R.
      colmap = terrain.colors(289))

# add the points to the persp plot.
perspPoints(species_df, Z = elevation, M = figure_3d, pch = 16, cex = 0.5)
```

```
## Warning: data contain duplicated points
```



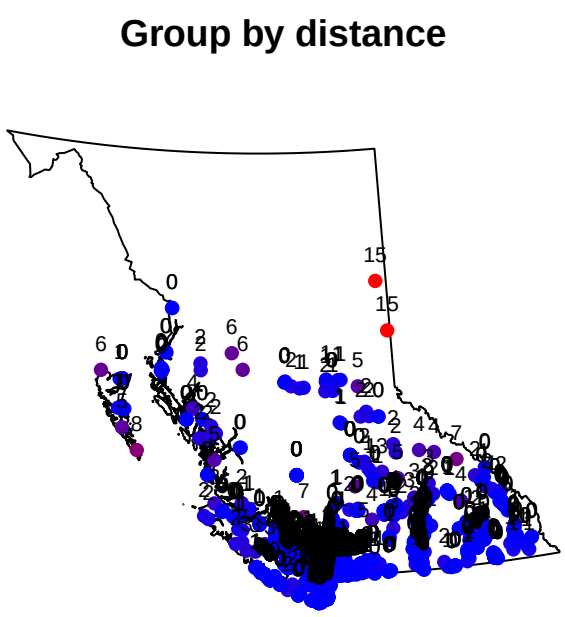
Elevation class for the specie

```
plot(cut(elevation, 5, labels = c("low", "median-low", "median", "median-high", "high")), main = "eleva

# add points
points(species_df$x, species_df$y, pch = 16, col = "black")
```



Group by distance

```
distances <- nndist(species_ppp)
species_ppp_with_marks <- species_ppp
marks(species_ppp_with_marks) <- data.frame(distance = distances)
col_pal <- colorRampPalette(c("blue", "red"))(100)
dist_scaled <- cut(distances, breaks = 100, labels = FALSE)
point_cols <- col_pal[dist_scaled]
plot(species_ppp, main = "", use.marks = FALSE, cex = 0.2)
title("Group by distance")
points(species_ppp$x, species_ppp$y, pch = 16, col = point_cols)
text(species_ppp$x, species_ppp$y, labels = round(distances/10000, 0), pos = 3, offset = 0.5, cex = 0.7)
```



Observation:

Most points are `dark blue` to `purple`, indicating the nearest neighbors are very close. Therefore, the specie tend to cluster in the south of the BC, and most of them are very close. However, the `red` points means that they are far away from their neighbors with `15`, indicating 150 km away from the neighbor.

Elevation Analysis

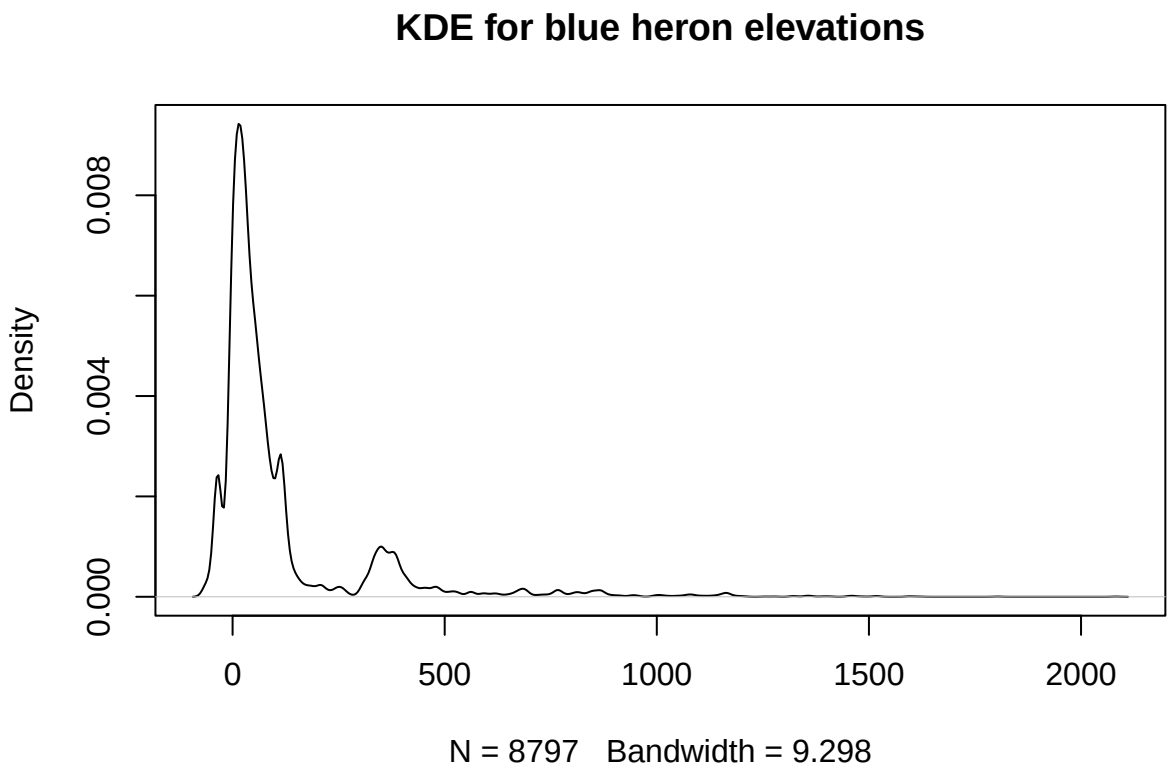
```
library(raster)
# 1. Convert spatstat::im (elevation) to raster
elev_raster <- raster(DATA$Elevation)
# get the elevation for specie
elevation_sp <- extract(elev_raster, cbind(species_df$x, species_df$y))
# get the median elevation from the park locations.
median_ele_sp <- median(elevation_sp, na.rm = TRUE)
cat("Median elevation of parks in BC is: ", median_ele_sp)
```

```
## Median elevation of parks in BC is:  38.8075
```

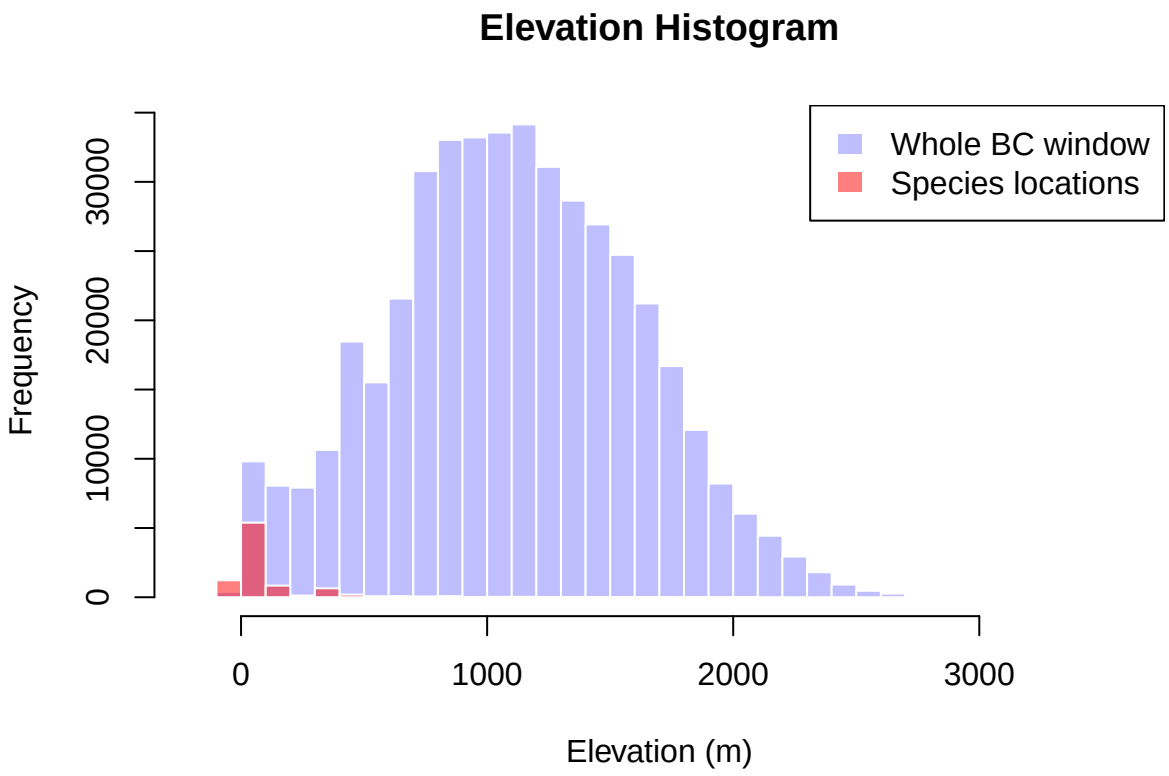
Conclusion:

We can see the specie reasonably live in the relatively low elevation area like `low-lying, flat areas`.

```r
kde_sp <- density(elevation_sp, na.rm = TRUE)
# plot the elevation density for the specie for better visualization
plot(kde_sp, main = "KDE for blue heron elevations")
```




```r
# Extract elevation values from the raster for the whole window (entire raster)
elev_all <- values(elev_raster)

# Extract elevation only at point locations
library(sp)
species_sp <- SpatialPoints(
  coords = data.frame(x = species_ppp$x, y = species_ppp$y),
  proj4string = CRS(projection(elev_raster))
)
elev_points <- extract(elev_raster, species_sp)

# Plot overlaid histograms
hist(elev_all, col = rgb(0, 0, 1, 0.25), main = "Elevation Histogram",
     xlab = "Elevation (m)", border = "white", breaks = 30)
hist(elev_points, col = rgb(1, 0, 0, 0.5), add = TRUE, border = "white", breaks = 30)
legend("topright", legend = c("Whole BC window", "Species locations"),
       fill = c(rgb(0,0,1,0.25), rgb(1,0,0,0.5)), border = NA)
```

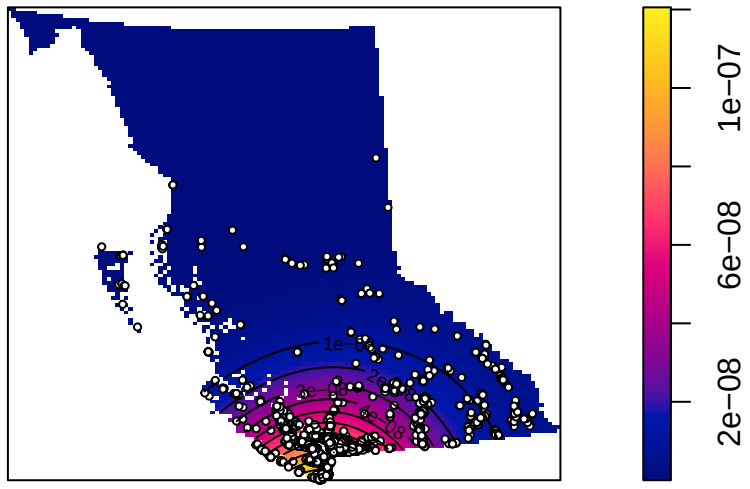


Conclusion:

Enhance the idea that **most of the specie in the low or median elevation area.**

Kernel-Density Distribution

```
density_map <- density(species_ppp)
plot(density_map, main = "Kernel Density of Species Observations")
points(species_ppp, pch = 21, cex = 0.5, col = "black", bg= "white")
contour(density_map, add = TRUE)
```



Conclusion:

Majority in the south corner and only few in the north.

Quadrat-Count based on elevation

```
#Extract elevation information
elev <- DATA$Elevation
#define quartiles
b <- quantile(elev, probs = (0:4)/4, type = 2)
#Split image into 4 equal-area quadrats based on elevation values
Zcut <- cut(elev, breaks = b)
V <- tess(image = Zcut)
quadratcount(species_ppp, tess = V)
```

```
## tile
##          (-130,761]       (761,1.1e+03]  (1.1e+03,1.46e+03] (1.46e+03,3.55e+03]
##                8848                 157                   48                  12
```

Apparently, most of them (8848) are located in low elevation

Quadrat Test

```
# make 10*10 quadrat
Q <- quadratcount(species_ppp, nx = 10, ny = 10)
# quadrat test
quadrat.test(Q)
```

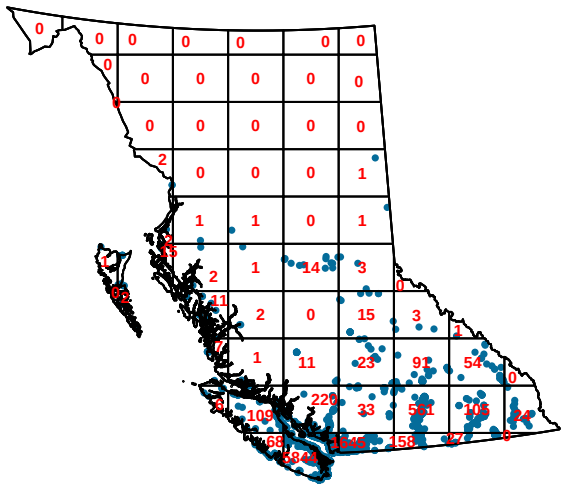
```
## Warning: Some expected counts are small; chi^2 approximation may be inaccurate
```

```
##
##  Chi-squared test of CSR using quadrat counts
##
## data:
## X2 = 317593, df = 63, p-value < 2.2e-16
## alternative hypothesis: two.sided
##
## Quadrats: 64 tiles (irregular windows)
```

Conclusion:

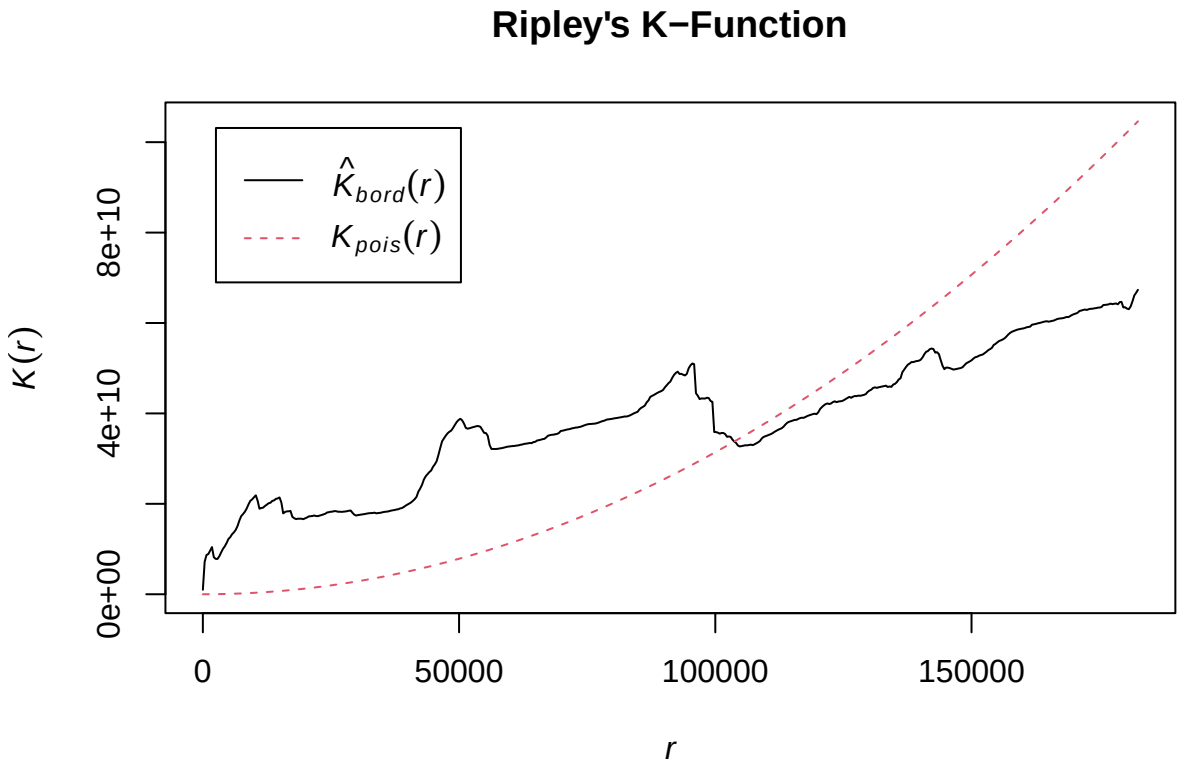
`p-value` small enough, so reject the null. **The assumption about homo is not met.**

```
Q <- quadratcount(species_ppp,
                  nx = 10,
                  ny = 10)

plot(species_ppp,
     pch = 16,
     cex = 0.5,
     cols = "#046C9A",
     main = "Beilschmiedia pendula locations")

plot(Q, cex = 0.5, col = "red", add = T, font = 2)
```



K-function

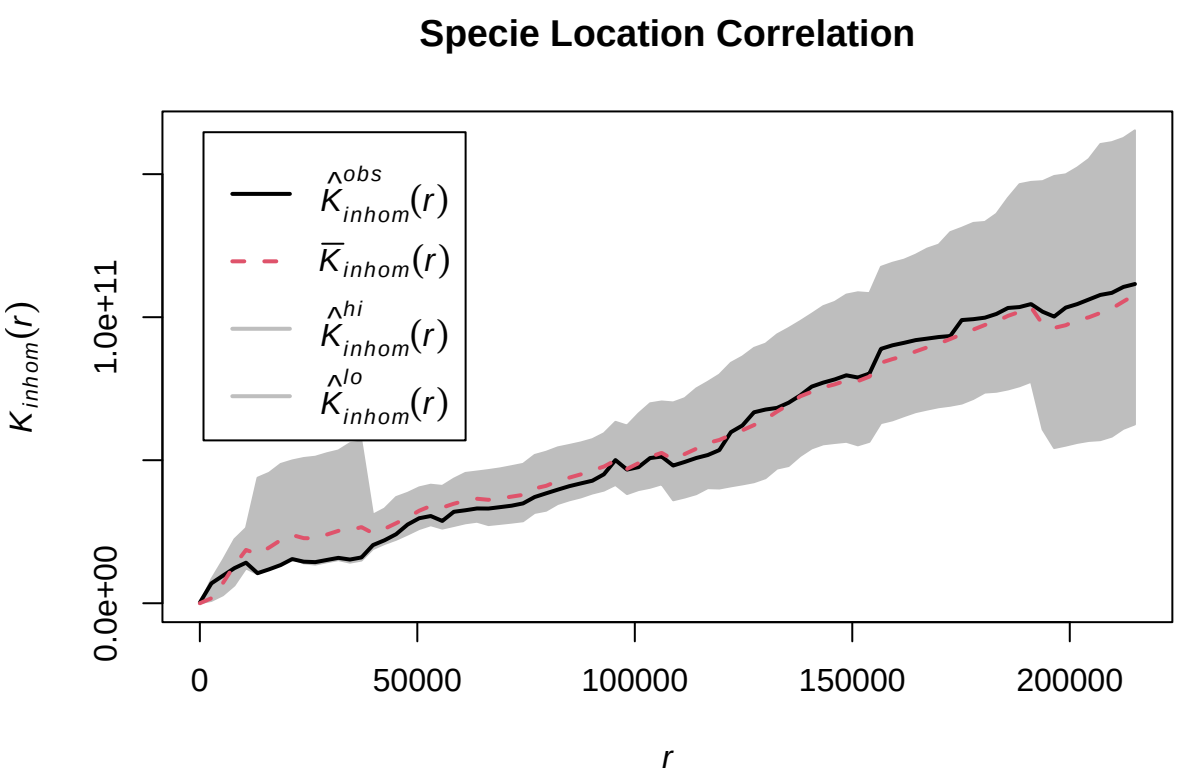
```
K <- Kest(species_ppp, correction = "border")
plot(K, main = "Ripley's K-Function")
```




```
lambda_sp_pos <- density(species_ppp,sigma=bw.ppl,positive=TRUE)

E_sp_inhom <- envelope(species_ppp,
                       Kinhom,
                       simulate = expression(rpoispp(lambda_sp_pos)),
                       correction="border",
                       rank = 1,
                       nsim = 19,
                       fix.n = TRUE)
```

```
## Warning in envelope.ppp(species_ppp, Kinhom, simulate =
## expression(rpoispp(lambda_sp_pos)), : fix.n and fix.marks were ignored, because
## 'simulate' was given

## Generating 19 simulations by evaluating expression  ...
## 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18,
## 19.
##
## Done.
```

```
plot(E_sp_inhom, main = "Specie Location Correlation", lwd = 2)
```



Observation:

The species locations appear randomly distributed, with no significant clustering or repulsion, because the `black line` is located within the CI.

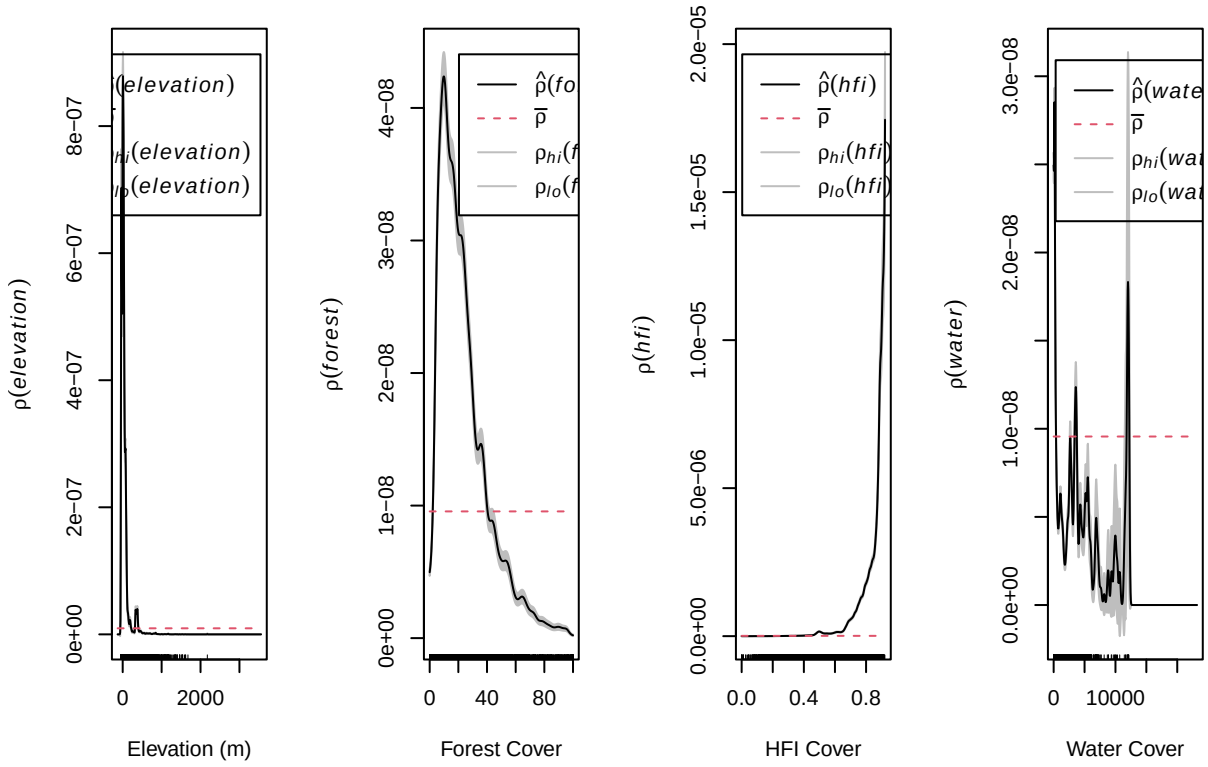
Check with the Covariates

```
elevation <- DATA$Elevation
forest <- DATA$Forest
# human footprint index
hfi <- DATA$HFI
water <- DATA$Dist_Water

rho_elev <- rhohat(species_ppp, elevation)
rho_forest <- rhohat(species_ppp, forest)
rho_hfi <- rhohat(species_ppp, hfi)
```

```
## Warning: Values for 4 query points lying outside the pixel image domain were
## estimated by projection to the nearest pixel
```

```
rho_water <- rhohat(species_ppp, water)

par(mfrow = c(1,4))
```

```
plot(rho_elev,
     main = "",
     xlab = "Elevation (m)")
plot(rho_forest,
     main = "",
     xlab = "Forest Cover")
plot(rho_hfi,
     main = "",
     xlab = "HFI Cover")
plot(rho_water,
     main = "",
     xlab = "Water Cover")
```



Conclusion:

- For the elevation, specie more located in the low elevation area.
- For the forest, specie tend to located in the low forest cover as well.
- For the HFI cover, specie surprisedly tend to live in the area have more human impact.
- For the water-distanced, the specie tend to live various water condition area, and the distinct peak is around the moderate water distance area.

```
cor.im(elevation, forest, hfi, water, use = "complete.obs")
```

```
##             ..1         ..2         ..3         ..4
## ..1  1.00000000 -0.26225376 -0.26625626 -0.03493453
## ..2 -0.26225376  1.00000000  0.06618592  0.04818598
## ..3 -0.26625626  0.06618592  1.00000000  0.13246899
## ..4 -0.03493453  0.04818598  0.13246899  1.00000000
```

```
fit <- ppm(species_ppp ~ elevation + I(elevation^2) + forest + I(forest^2) + hfi + I(hfi^2) + water + I
```

```
## Warning in countingweights(id, areas): some tiles with positive area do not
## contain any quadrature points: relative error = 1.3%
```

```
## Warning: Values of the covariates 'elevation', 'hfi' were NA or undefined at
## 0.42% (111 out of 26433) of the quadrature points. Occurred while executing:
## ppm.ppp(Q = species_ppp, trend = ~elevation + I(elevation^2) +
```

```
fit
```

```
## Error in solve.default(M) :
##   system is computationally singular: reciprocal condition number = 8.02341e-18
```

```
## Warning: Cannot compute variance: Fisher information matrix is singular
```

```
## Error in solve.default(M) :
##   system is computationally singular: reciprocal condition number = 8.02341e-18
```

```
## Warning: Cannot compute variance: Fisher information matrix is singular
```

```
## Nonstationary Poisson process
## Fitted to point pattern dataset 'species_ppp'
##
## Log intensity:  ~elevation + I(elevation^2) + forest + I(forest^2) + hfi +
## I(hfi^2) + water + I(water^2)
##
## Fitted trend coefficients:
##    (Intercept)      elevation I(elevation^2)         forest    I(forest^2)
##  -1.790835e+01  -6.324162e-03   1.729139e-06  -2.561233e-02   1.696736e-04
##            hfi       I(hfi^2)          water     I(water^2)
##   1.015077e+01  -3.827161e+00  -2.651529e-04   3.861176e-09
##
## Standard errors unavailable; Fisher information matrix is singular
## Problem:
##  Values of the covariates 'elevation', 'hfi' were NA or undefined at 0.42% (111
## out of 26433) of the quadrature points
```

```
fit0 <- ppm(species_ppp ~ 1)
```

```
## Warning in countingweights(id, areas): some tiles with positive area do not
## contain any quadrature points: relative error = 1.3%
```

```r
fit0
```

```
## Stationary Poisson process
## Fitted to point pattern dataset 'species_ppp'
## Intensity: 9.55959e-09
##              Estimate       S.E.   CI95.lo   CI95.hi Ztest      Zval
## log(lambda) -18.46572 0.01050307 -18.48631 -18.44514   *** -1758.127
```

```r
AIC(fit)
```

```
## [1] 269901.1
```

```r
AIC(fit0)
```

```
## [1] 352915.5
```

```r
anova(fit, fit0, test = "LRT")
```

```
## Warning in countingweights(id, areas): some tiles with positive area do not
## contain any quadrature points: relative error = 1.3%
```

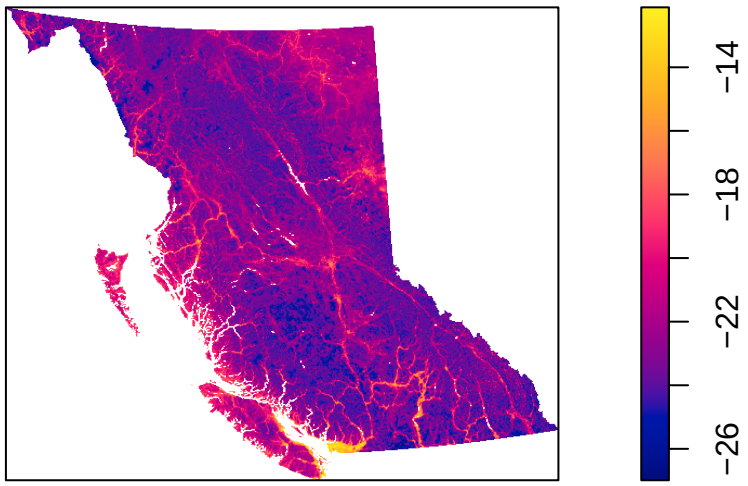
```
## Warning: Values of the covariates 'elevation', 'hfi' were NA or undefined at
## 0.42% (111 out of 26433) of the quadrature points. Occurred while executing:
## ppm.ppp(Q = species_ppp, trend = ~elevation + I(elevation^2) +
```

```
## Warning in countingweights(id, areas): some tiles with positive area do not
## contain any quadrature points: relative error = 1.3%
```

```
## Warning: Models were re-fitted after discarding quadrature points that were
## illegal under some of the models
```

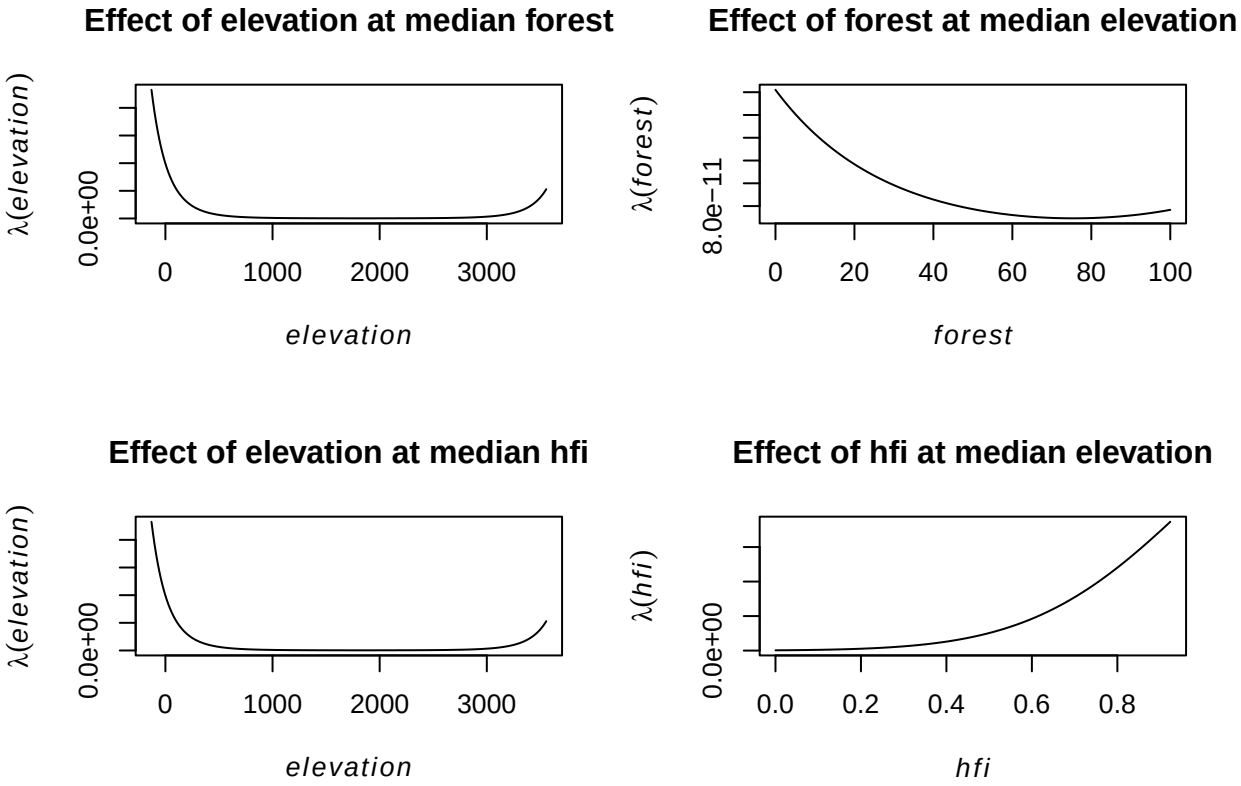
```
## Analysis of Deviance Table
##
## Model 1: ~elevation + I(elevation^2) + forest + I(forest^2) + hfi + I(hfi^2) + water + I(water^2)
## Model 2: ~1   Poisson
##   Npar Df Deviance  Pr(>Chi)
## 1    9
## 2    1 -8   -80056 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

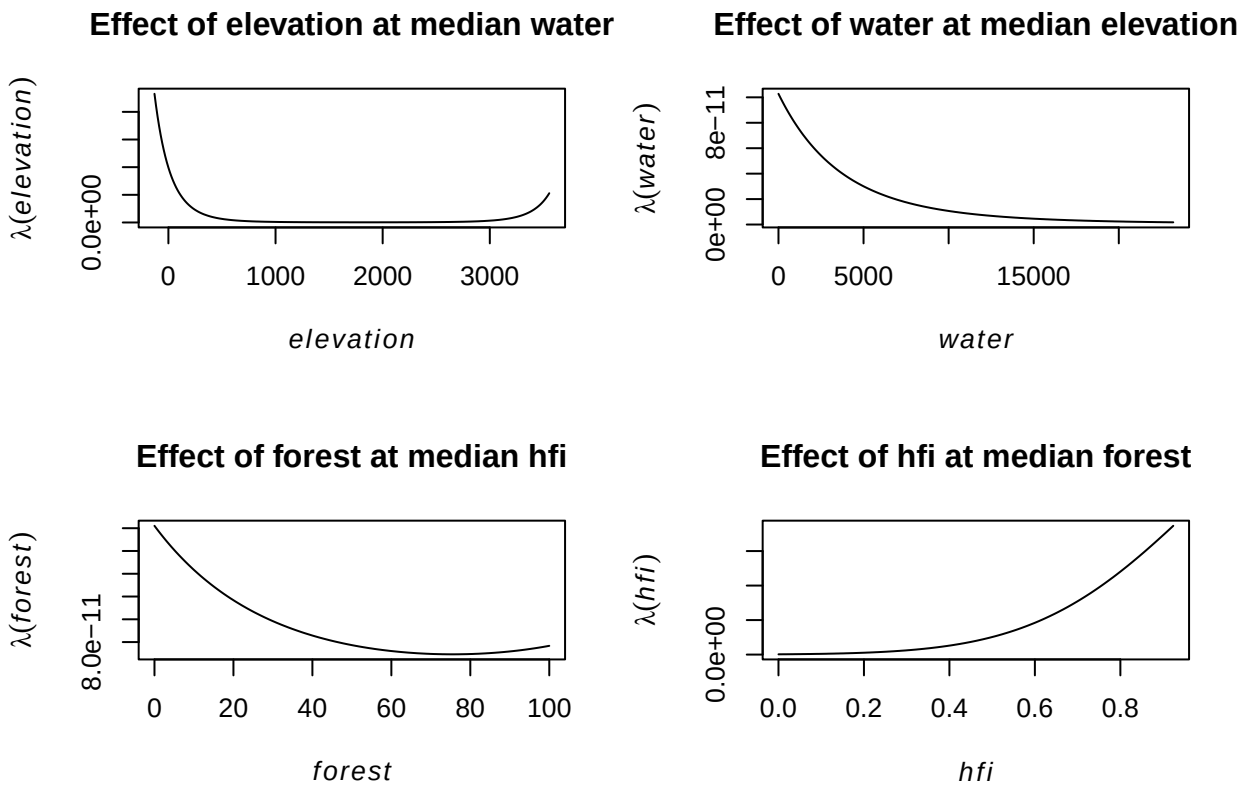
```r
predicted <- predict(fit, type = "trend", n = 512)

log_intensity <- eval.im(log(predicted))

plot(log_intensity,
     main = "Log-Scaled Fitted Model Intensity",
     se = FALSE,
     superimpose = FALSE)
```

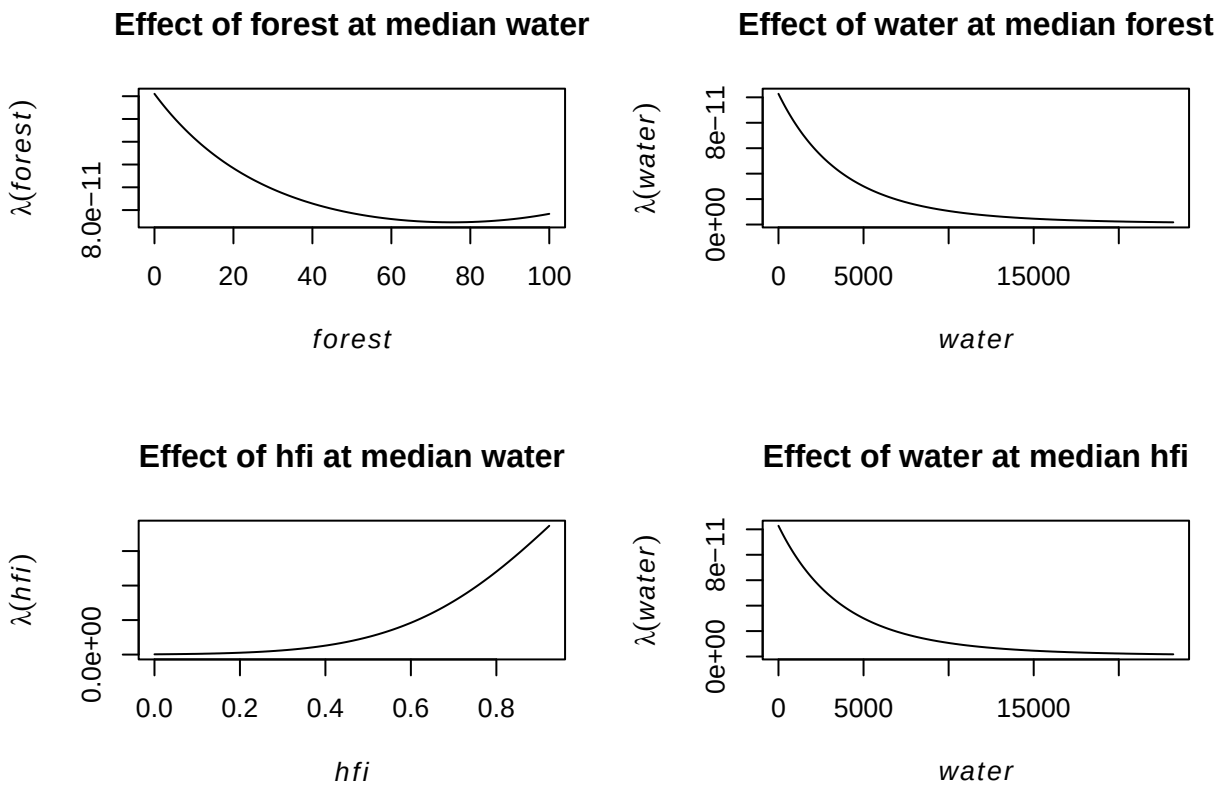



```
med_elev   <-  median(DATA$Elevation$v,   na.rm=TRUE)
med_forest <-  median(DATA$Forest$v,      na.rm=TRUE)
med_hfi    <-  median(DATA$HFI$v,         na.rm=TRUE)
med_water  <-  median(DATA$Dist_Water$v,  na.rm=TRUE)

cov_medians <- list(
  elevation       = med_elev,
  `I(elevation^2)` = med_elev^2,
  forest          = med_forest,
  `I(forest^2)`    = med_forest^2,
  hfi             = med_hfi,
  `I(hfi^2)`       = med_hfi^2,
  water           = med_water,
  `I(water^2)`     = med_water^2
)

vars  <- c("elevation","forest","hfi","water")
pairs <- combn(vars, 2, simplify = FALSE)

oldpar <- par(mfrow = c(2, 2))

for(pair in pairs) {
  v1 <- pair[1]
  v2 <- pair[2]

  args1 <- cov_medians
```

```
  args1[[v1]]                      <- NULL
  args1[[paste0("I(",v1,"^2)")]]   <- NULL

  ef1 <- do.call(
    effectfun,
    c(list(fit, covname = v1, se.fit = FALSE),
      args1)
  )
  plot(ef1,
       legend = FALSE,
       ylab   = bquote(lambda(.(as.name(v1)))),
       main   = paste("Effect of", v1, "at median", v2))

  args2 <- cov_medians
  args2[[v2]]                      <- NULL
  args2[[paste0("I(",v2,"^2)")]]   <- NULL

  ef2 <- do.call(
    effectfun,
    c(list(fit, covname = v2, se.fit = FALSE),
      args2)
  )
  plot(ef2,
       legend = FALSE,
       ylab   = bquote(lambda(.(as.name(v2)))),
       main   = paste("Effect of", v2, "at median", v1))
}
```


Effect of elevation at median forest
λ(elevation)
0.0e+00
0
1000
2000
3000
elevation
Effect of forest at median elevation
λ(forest)
8.0e−11
0
20
40
60
80
100
forest
Effect of elevation at median hfi
λ(elevation)
0.0e+00
0
1000
2000
3000
elevation
Effect of hfi at median elevation
λ(hfi)
0.0e+00
0.0
0.2
0.4
0.6
0.8
hfi


Effect of elevation at median water
λ(elevation)
0.0e+00
0
1000
2000
3000
elevation
Effect of water at median elevation
λ(water)
0e+00
8e−11
0
5000
15000
water
Effect of forest at median hfi
λ(forest)
8.0e−11
0
20
40
60
80
100
forest
Effect of hfi at median forest
λ(hfi)
0.0e+00
0.0
0.2
0.4
0.6
0.8
hfi




```
par(oldpar)
```

```
par_res_elev <- parres(fit, "elevation")
```

```
## Warning: Some infinite, NA or NaN increments were removed
```

```
par_res_forest <- parres(fit, "forest")
```

```
## Warning: Some infinite, NA or NaN increments were removed
```

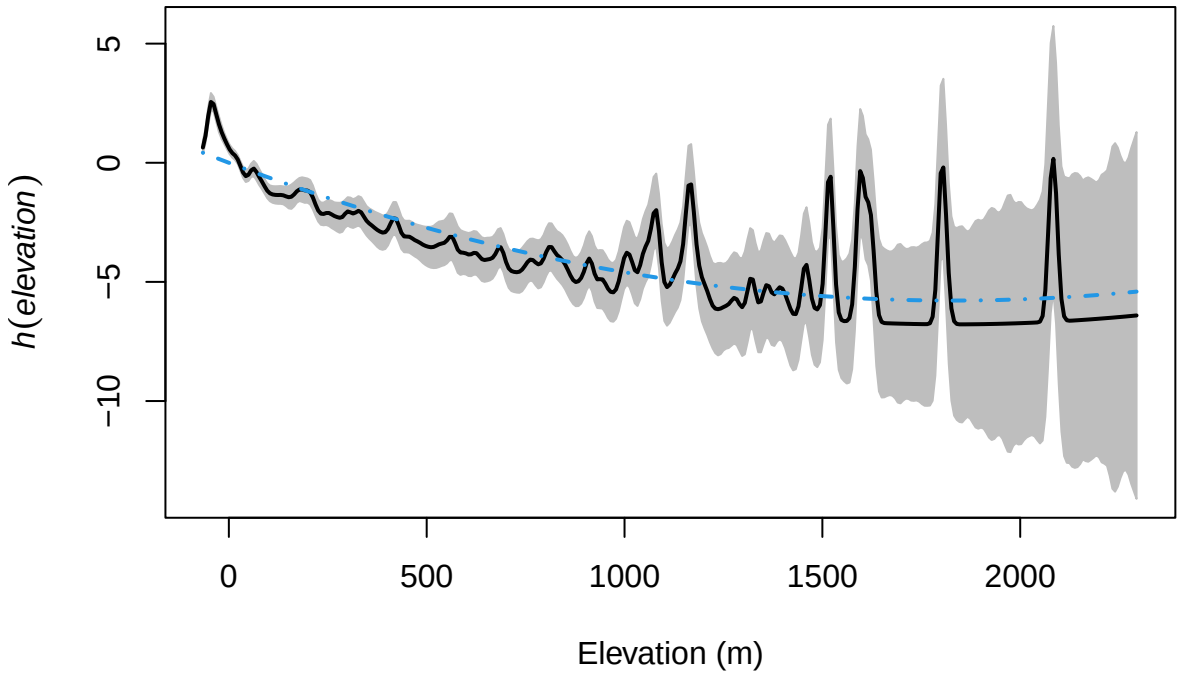
```
par_res_hfi <- parres(fit, "hfi")
```

```
## Warning: Some infinite, NA or NaN increments were removed

## Warning: Values for 14 query points lying outside the pixel image domain were
## estimated by projection to the nearest pixel
```

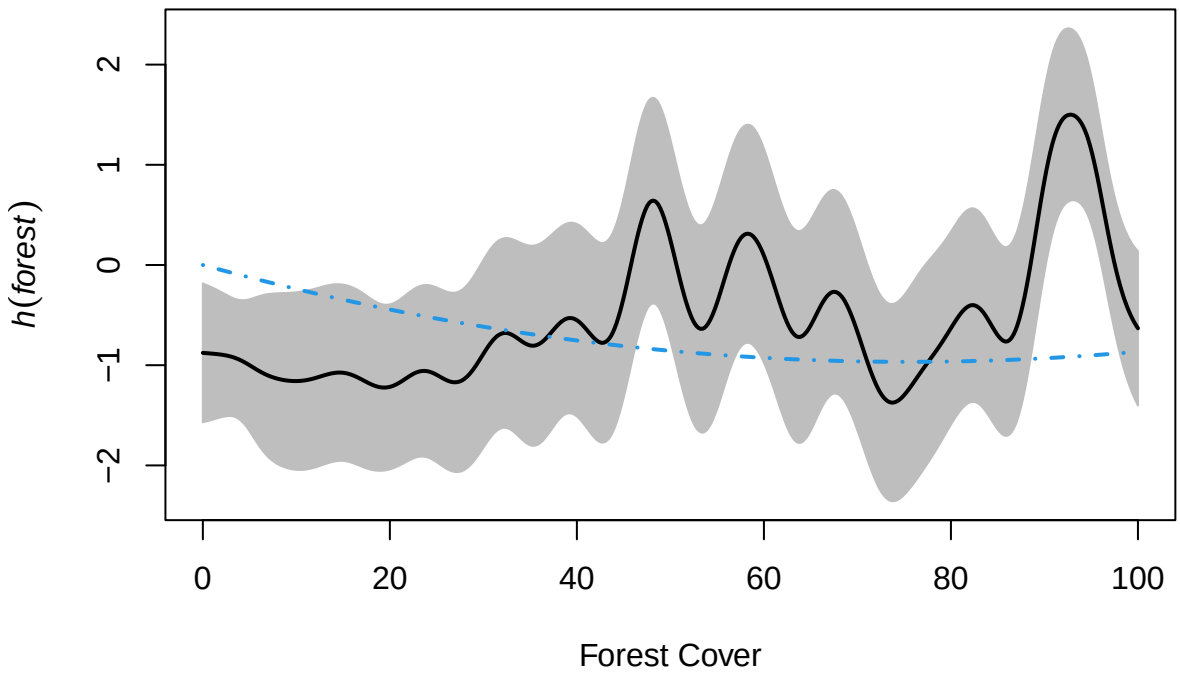
```
par_res_water <- parres(fit, "water")
```

```
## Warning: Some infinite, NA or NaN increments were removed
```

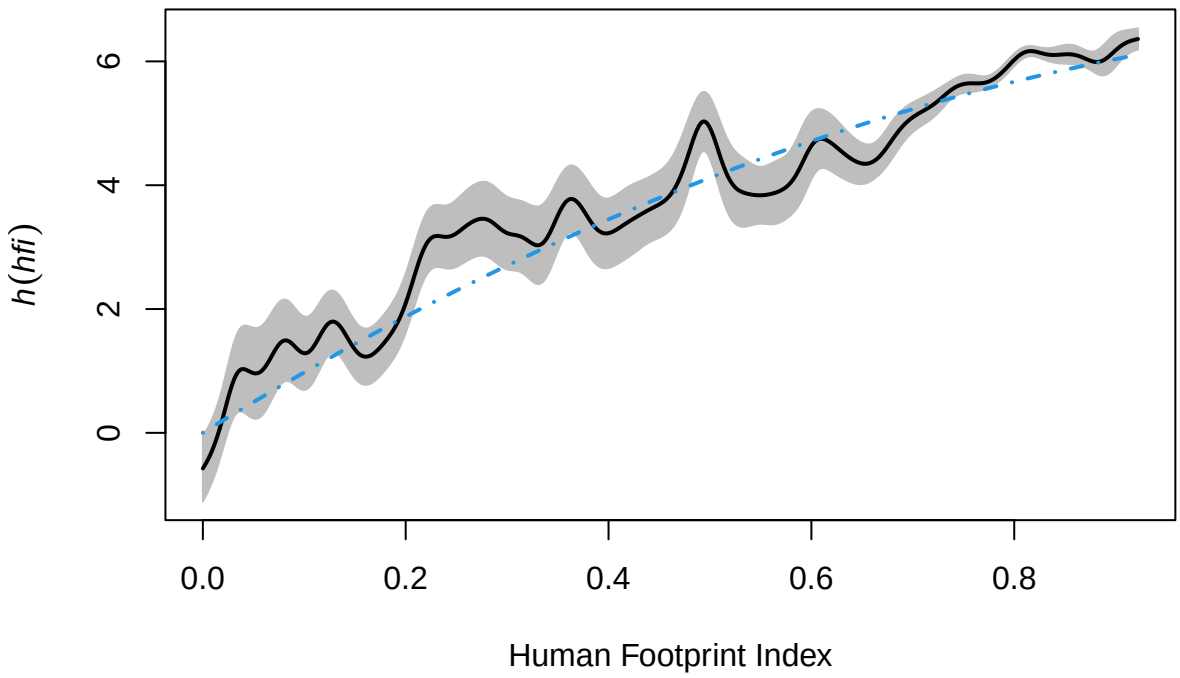
```r
plot(par_res_elev,
     legend = FALSE,
     lwd = 2,
     main = "",
     xlab = "Elevation (m)")
```



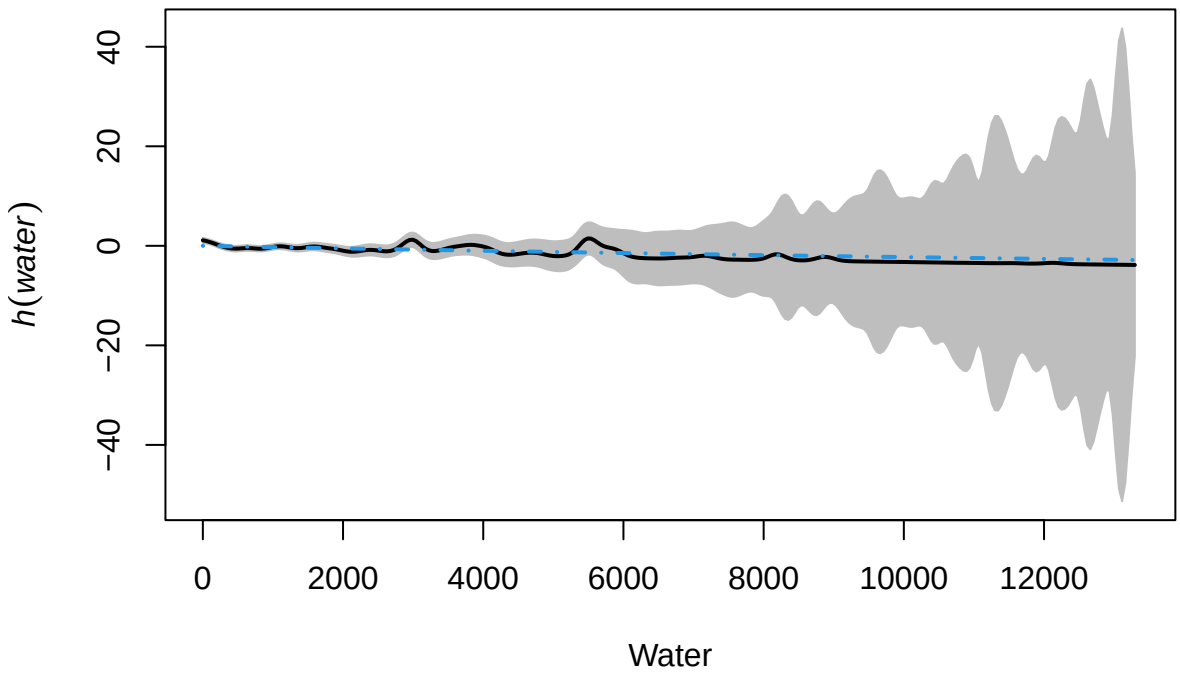

```r
plot(par_res_forest,
     legend = FALSE,
     lwd = 2,
     main = "",
     xlab = "Forest Cover")
```




```
plot(par_res_hfi,
     legend = FALSE,
     lwd = 2,
     main = "",
     xlab = "Human Footprint Index")
```




```
plot(par_res_water,
     legend = FALSE,
     lwd = 2,
     main = "",
     xlab = "Water")
```


h(water)
40
20
0
−20
−40
0
2000
4000
6000
8000
10000
12000
Water