Project

Tianyi Xia

10 April, 2025 16:10:50

Demo for env var

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

```
## Loading required package: spatstat.data
## Loading required package: spatstat.univar
## spatstat.univar 3.1-2
## Loading required package: spatstat.geom
## spatstat.geom 3.3-6
## Loading required package: spatstat.random
## spatstat.random 3.3-3
## Loading required package: spatstat.explore
## Loading required package: nlme
## spatstat.explore 3.4-2
## Loading required package: spatstat.model
## Loading required package: rpart
## spatstat.model 3.3-5
## Loading required package: spatstat.linnet
## spatstat.linnet 3.2-5
##
## spatstat 3.3-2
## For an introduction to spatstat, type 'beginner'
```

```
library(sf)
```

```
## Linking to GEOS 3.13.0, GDAL 3.8.5, PROJ 9.5.1; sf_use_s2() is TRUE
```

```
library(raster)
```

```
## Loading required package: sp
##
## Attaching package: 'raster'
## The following object is masked from 'package:nlme':
##
```

```
##     getData
```

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

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

```
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
```

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

```
# 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
```

```
load("BC_covariates.Rda")
window <- DATA$Window
class(DATA$Window)
```

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

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

```
# 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
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_coordiantes <- transformed_coords[,1]
y_coordinates <- transformed_coords[,2]
species_ppp <- ppp(
  x = x_coordiantes,
  y = y_coordinates,
  window = window_owin
```

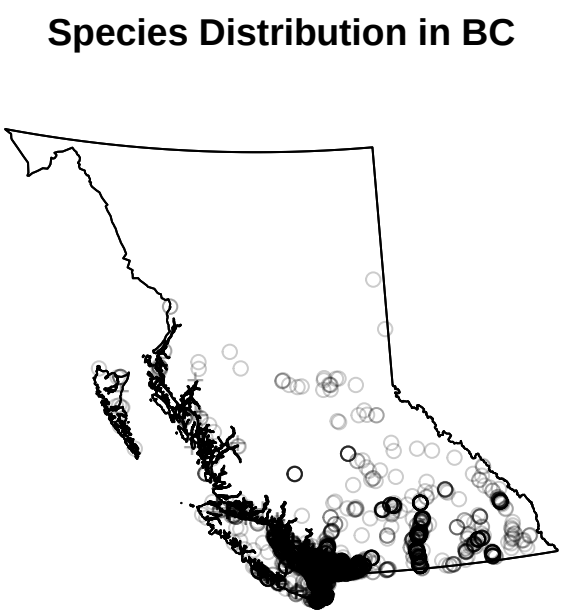
```
)
```

```
## Warning: 1759 points were rejected as lying outside the specified window
```

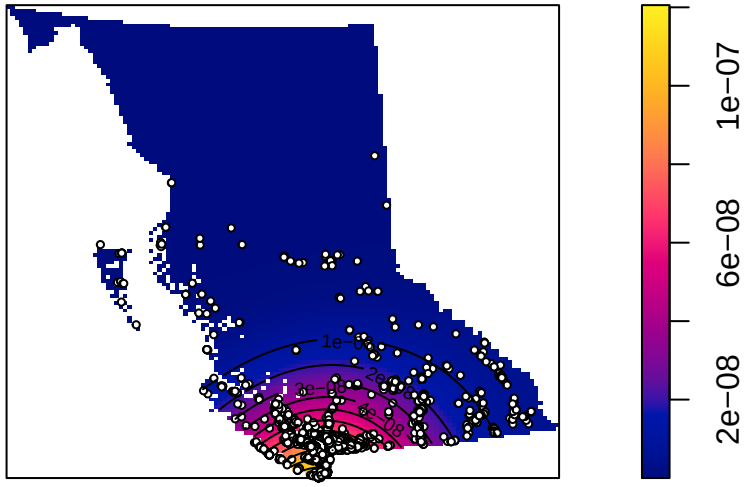
```
## Warning: data contain duplicated points
```

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

```
## Warning in plot.ppp(species_ppp, main = "Species Distribution in BC"): 1759
## illegal points also plotted
```



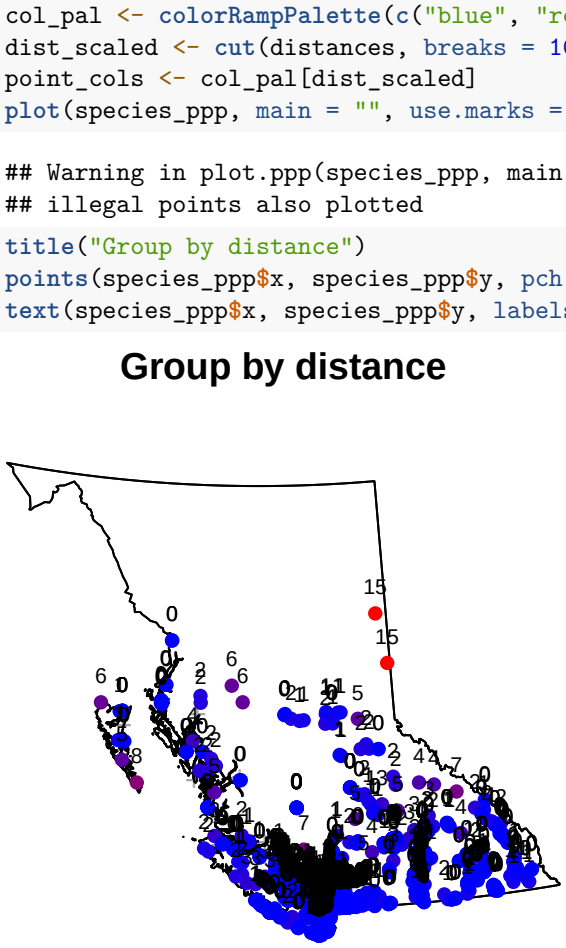

```
density_map <- density(species_ppp)
plot(density_map, main = "Kernel Density of Species Observations")
points(species_ppp, pch = 21, cex = 0.5, col = "black", bg= "white")
contour(density_map, add = TRUE)
```



```
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)
```

```
## Warning in plot.ppp(species_ppp, main = "", use.marks = FALSE, cex = 0.2): 1759
## illegal points also plotted
```

```
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)
```



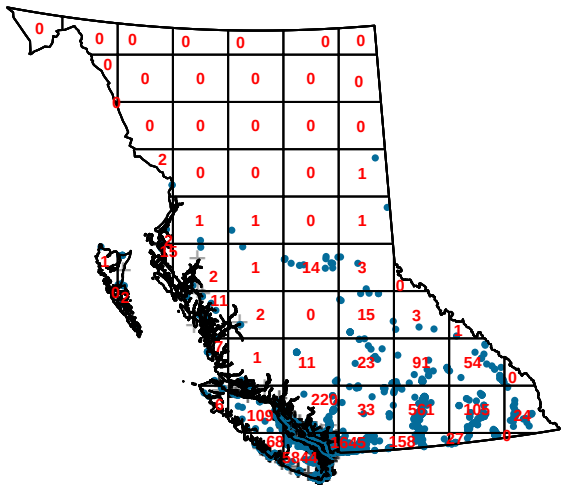

```
Q <- quadratcount(species_ppp,
                  nx = 10,
                  ny = 10)

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

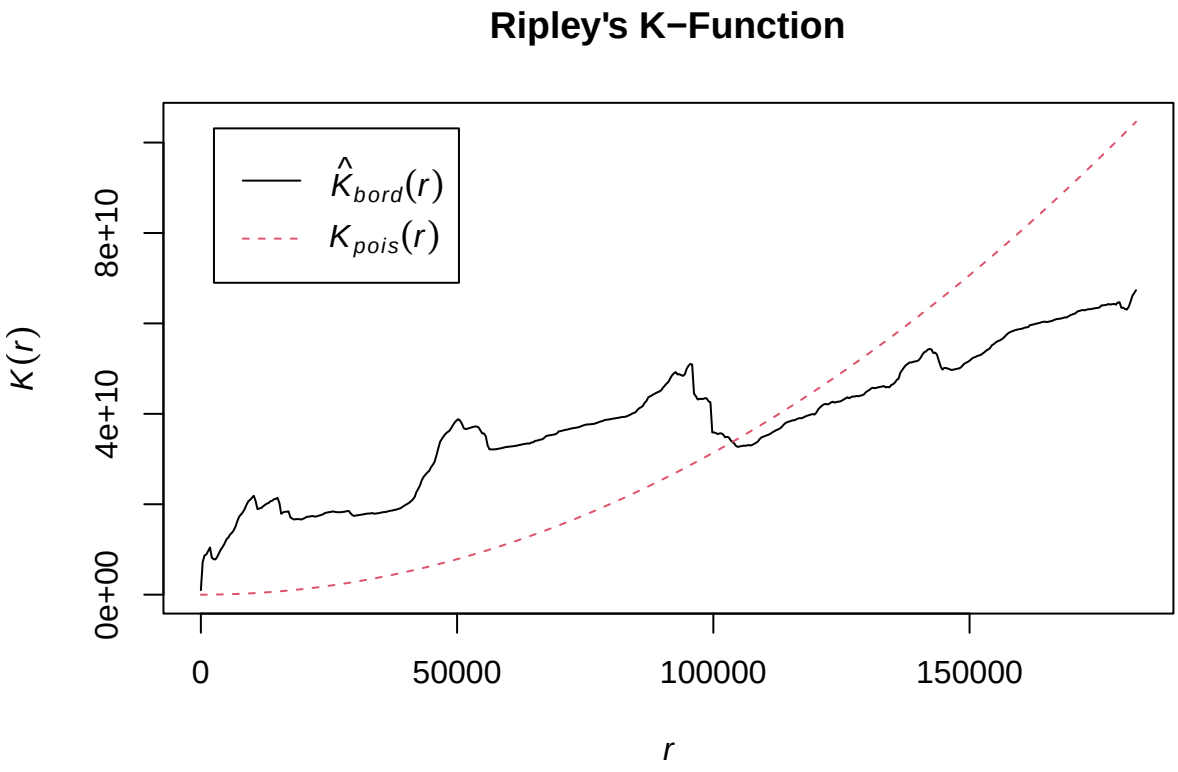
```
## Warning in plot.ppp(species_ppp, pch = 16, cex = 0.5, cols = "#046C9A", : 1759
## illegal points also plotted
```

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

Beilschmiedia pendula locations



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




```
library(raster)
# 1. Convert spatstat::im (elevation) to raster
elev_raster <- raster(DATA$Elevation)

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

# 3. Extract elevation only at point locations
```

```r
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)

# 4. 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)
```