Seedlings dataset

```r
library(ppjsdm)
#> Registered S3 method overwritten by 'spatstat':
#>   method     from
#>   print.boxx cli
library(ecespa)
#> Loading required package: spatstat
#> Loading required package: spatstat.data
#> Loading required package: nlme
#> Loading required package: rpart
#>
#> spatstat 1.64-0       (nickname: 'Susana Distancia')
#> For an introduction to spatstat, type 'beginner'
remove(list = ls())

set.seed(1)
```

This vignette explains how to use the `ppjsdm` package with the `seedlings2` dataset from `ecespa`. This dataset is a marked point pattern of seedlings of H. squamatumgrowing in a gyp-sophylous plant community in Central Spain. The datasets contains the locations of the seedlings marked with their heigth. Both the coordinates and the heigth of the seedlings are given in cm.

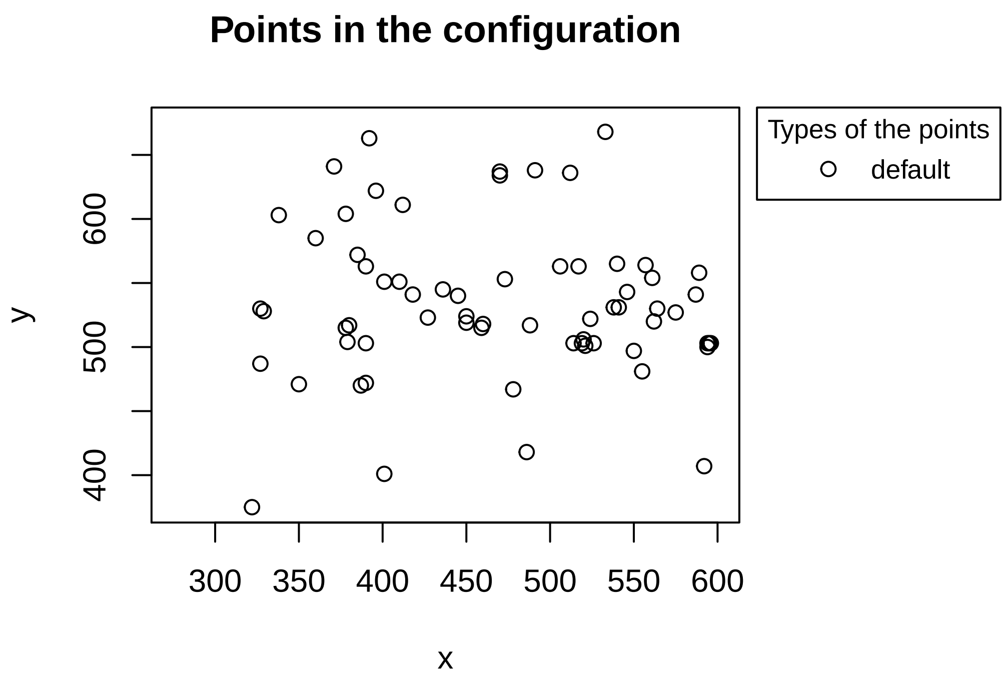
Taking marks into account

If marks (i.e. their height) are provided, the interaction radii are proportional to the marks. We begin with that setting.

```r
configuration <- Configuration(seedlings2$x, seedlings2$y, marks = seedlings2$marks)
window <- Rectangle_window(c(275, 600), c(375, 675))
```

The point configuration is plotted below.

```r
print(configuration)
#> An S3 object representing a configuration.
#>
#> Number of points: 66.
par(mar = c(5, 4, 4, 13) + 0.1)
plot(configuration, window = window)
```



We provide a series of ranges for the interaction radii, and let the fitting function calibrate the model.

```
short_range <- c(0, 20)
medium_range <- c(0, 20)
long_range <- c(0, 20)
model <- "square_bump"
medium_range_model <- "square_exponential"
saturation <- 2
steps <- 100000
```

We can now call the fitting function.

```
fit <- ppjsdm::gibbsm(configuration,
                      window = window,
                      model = model,
                      medium_range_model = medium_range_model,
                      short_range = short_range,
                      medium_range = medium_range,
                      long_range = long_range,
                      use_glmnet = FALSE,
                      use_aic = TRUE,
                      saturation = saturation)
#> $beta0
#> [1] -7.426361
#>
#> $alpha
#>           [,1]
#> [1,] 0.5875153
#>
#> $gamma
#>            [,1]
#> [1,] -0.5035048
#> 
#> $beta
#>     
#> [1,]
print(summary(fit))
#>             coefficients        se      CI95_lo     CI95_hi Ztest       Pval
#> log_lambda1   -7.4263613 0.1799016 -7.778961895 -7.07376065   *** 0.00000000
#> alpha_1_1      0.5875153 0.2979012  0.003639712  1.17139099     * 0.04858885
#> gamma_1_1     -0.5035048 0.2709669 -1.034590185  0.02758064       0.06314374
#>                   Zval
#> log_lambda1 -41.280133
#> alpha_1_1     1.972182
#> gamma_1_1    -1.858178
print(fit$coefficients)
#> $beta0
#> [1] -7.426361
#> 
#> $alpha
#>           [,1]
#> [1,] 0.5875153
#> 
#> $gamma
#>            [,1]
#> [1,] -0.5035048
#> 
#> $beta
#>     
#> [1,]
#> 
#> $short_range
#>          [,1]
#> [1,] 4.605498
#> 
#> $medium_range
#>          [,1]
#> [1,] 7.226139
#> 
#> $long_range
#>         [,1]
#> [1,] 9.97959
print(fit$aic)
#> [1] 398.1983
print(fit$bic)
#> [1] 411.2141
```

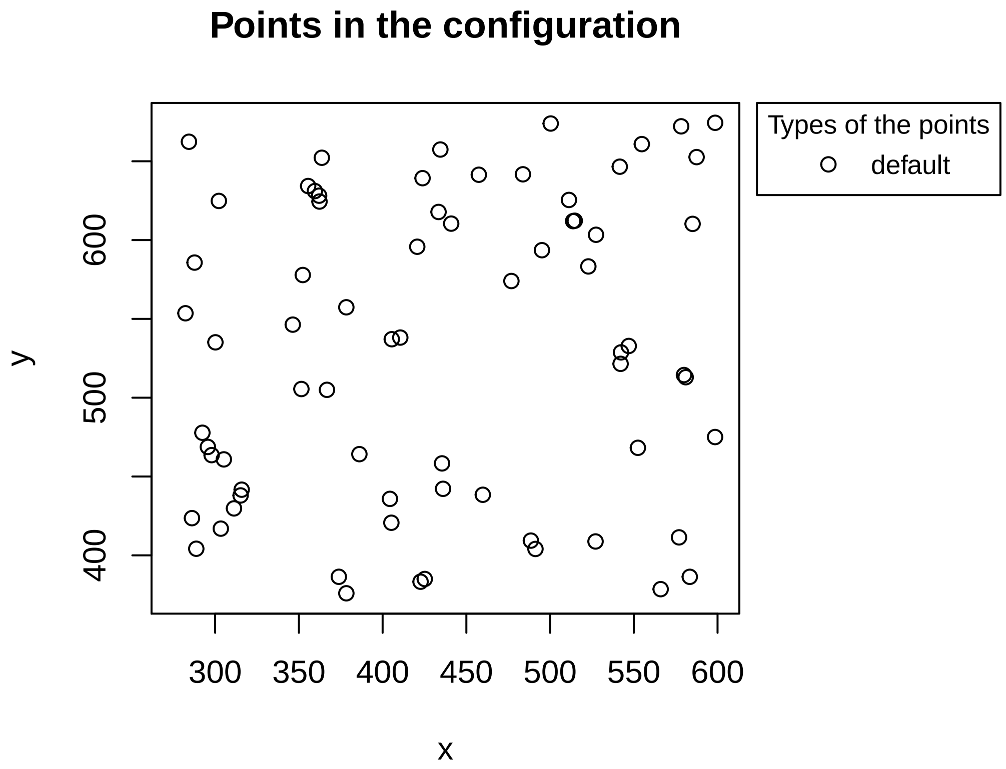
We may then plot the corresponding Papangelou conditional intensity.

```
parameters <- fit$coefficients
# plot_papangelou(window = window,
#                 configuration = configuration,
#                 type = 1,
#                 mark = mean(get_marks(configuration)),
#                  model = model,
#                  medium_range_model = medium_range_model,
#                  alpha = parameters$alpha,
#                  beta0 = parameters$beta0,
#                  beta = matrix(0, 1, 0),
#                  gamma = parameters$gamma,
#                  covariates = list(),
#                  short_range = parameters$short_range,
#                  medium_range = parameters$medium_range,
#                  long_range = parameters$long_range,
#                  saturation = saturation)
```

It is also possible to draw from the model.

```
draw <- ppjsdm::rgibbs(window = window,
                       alpha = parameters$alpha,
                       beta0 = parameters$beta0,
                       gamma = parameters$gamma,
                       model = model,
                       medium_range_model = medium_range_model,
                       short_range = parameters$short_range,
                       medium_range = parameters$medium_range,
                       long_range = parameters$long_range,
                       types = levels(types(configuration)),
                       mark_range = c(min(get_marks(configuration)), max(get_marks(configuration))),
                       steps = steps)
print(draw)
#> An S3 object representing a configuration.
#>
#> Number of points: 71.

par(mar = c(5, 4, 4, 13) + 0.1)
plot(draw, window = window)
```



Interaction radii not proportional to marks

In this section, we disregard the height of the seedlings, and consider interaction radii in centimetres.

```
configuration <- Configuration(seedlings2$x, seedlings2$y)
```

We call the fitting function on this unmarked point process.

```
fit <- ppjsdm::gibbsm(configuration,
                      window = window,
                      model = model,
                      medium_range_model = medium_range_model,
                      short_range = short_range,
                      medium_range = medium_range,
                      long_range = long_range,
                      use_glmnet = FALSE,
                      use_aic = TRUE,
                      saturation = saturation)
#> $beta0
#> [1] -7.932892
#>
#> $alpha
#>          [,1]
#> [1,] 1.224693
#>
```

```
#> $gamma
#>           [,1]
#> [1,] 0.0386277
#> 
#> $beta
#>     
#> [1,]
print(summary(fit))
#>             coefficients        se     CI95_lo     CI95_hi Ztest         Pval
#> log_lambda1   -7.9328921 0.1895316 -8.3043672 -7.5614169   *** 0.000000e+00
#> alpha_1_1      1.2246925 0.2670578  0.7012688  1.7481162   *** 4.520983e-06
#> gamma_1_1      0.0386277 0.2113301 -0.3755717  0.4528271       8.549677e-01
#>                    Zval
#> log_lambda1 -41.8552444
#> alpha_1_1     4.5858707
#> gamma_1_1     0.1827837
print(fit$coefficients)
#> $beta0
#> [1] -7.932892
#> 
#> $alpha
#>          [,1]
#> [1,] 1.224693
#> 
#> $gamma
#>           [,1]
#> [1,] 0.0386277
#> 
#> $beta
#>     
#> [1,]
#> 
#> $short_range
#>          [,1]
#> [1,] 6.181326
#> 
#> $medium_range
#>          [,1]
#> [1,] 18.61553
#> 
#> $long_range
#>          [,1]
#> [1,] 20.53795
print(fit$aic)
#> [1] 371.7991
print(fit$bic)
#> [1] 384.8149
```

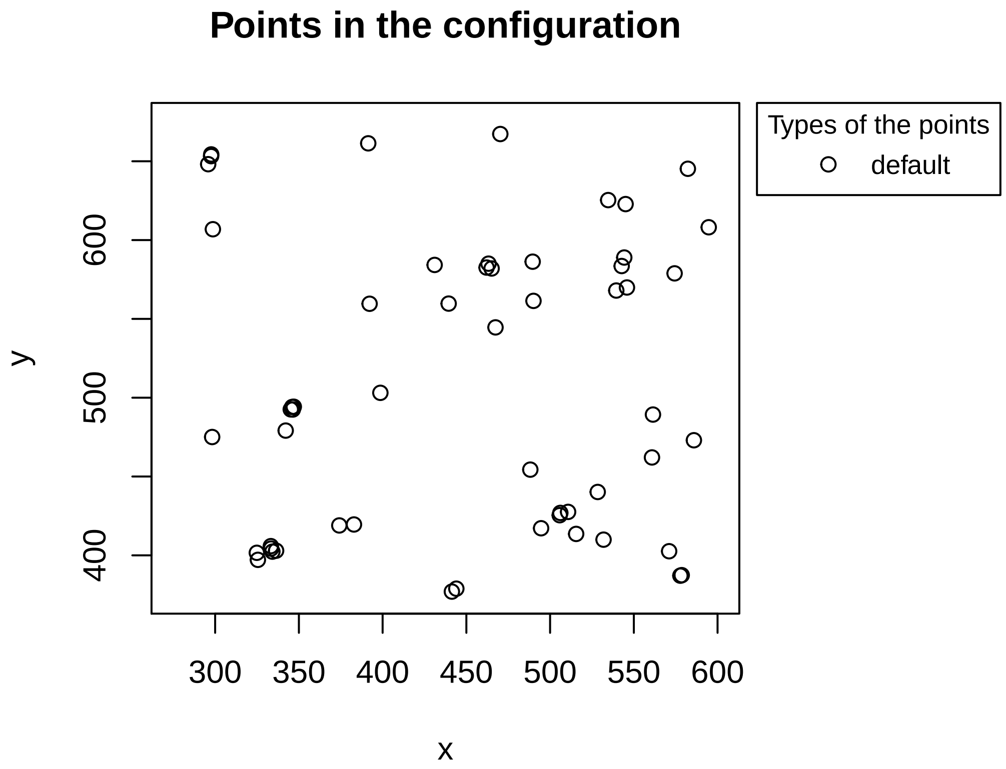
We may then plot the corresponding Papangelou conditional intensity.

```
parameters <- fit$coefficients
# plot_papangelou(window = window,
#                 configuration = configuration,
#                 type = 1,
#                 mark = mean(get_marks(configuration)),
#                 model = model,
#                 medium_range_model = medium_range_model,
#                 alpha = parameters$alpha,
#                 beta0 = parameters$beta0,
#                 beta = matrix(0, 1, 0),
#                 gamma = parameters$gamma,
#                 covariates = list(),
#                 short_range = parameters$short_range,
#                 medium_range = parameters$medium_range,
#                 long_range = parameters$long_range,
#                 saturation = saturation)
```

And as previously, we draw from the model.

```
draw <- ppjsdm::rgibbs(window = window,
                       alpha = parameters$alpha,
                       beta0 = parameters$beta0,
                       gamma = parameters$gamma,
                       model = model,
                       medium_range_model = medium_range_model,
                       short_range = parameters$short_range,
                       medium_range = parameters$medium_range,
                       long_range = parameters$long_range,
                       types = levels(types(configuration)),
                       mark_range = c(min(get_marks(configuration)), max(get_marks(configuration))),
                       steps = steps)
print(draw)
#> An S3 object representing a configuration.
#>
#> Number of points: 55.

par(mar = c(5, 4, 4, 13) + 0.1)
plot(draw, window = window)
```


Points in the configuration
Types of the points
○ default
y
600
500
400
300
350
400
450
500
550
600
x