Fin pines dataset

```
library(ppjsdm)
#> Registered S3 method overwritten by 'spatstat':
#>   method     from
#>   print.boxx cli
library(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 `finpines` dataset from `spatstat`. The data record the locations of 126 pine saplings in a Finnish forest, their heights and their diameters.

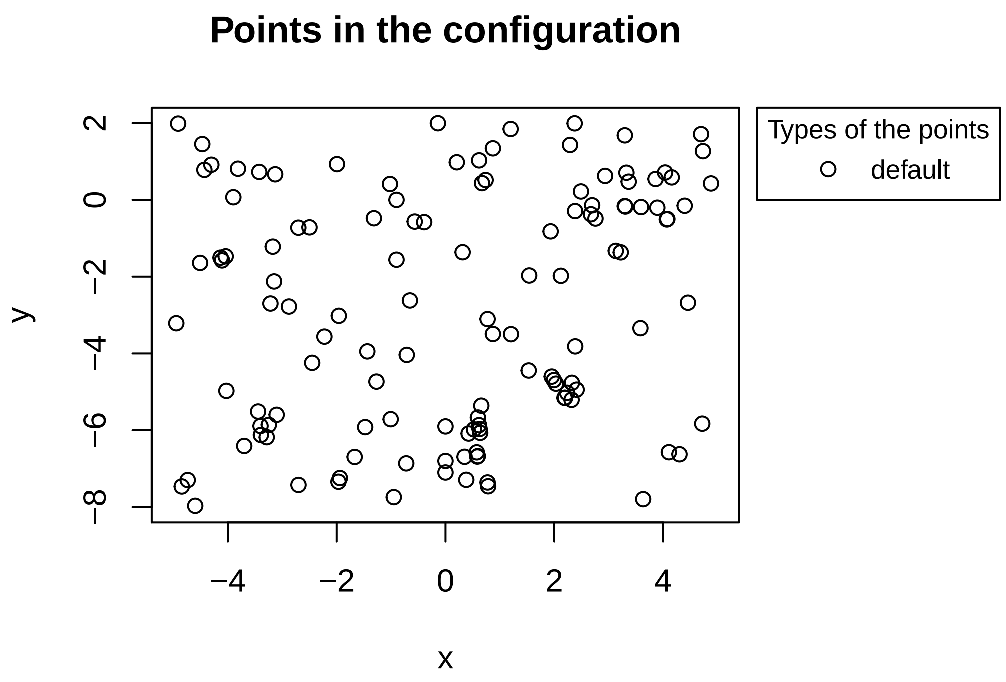
Taking marks into account

If marks are provided, the interaction radii are proportional to the marks. We begin with that setting, choosing the height (rather than the diameter) as the relevant mark.

```
configuration <- Configuration(finpines$x, finpines$y, marks = finpines$marks$height)
window <- Rectangle_window(c(-5, 5), c(-8, 2))
```

The point configuration is plotted below.

```
print(configuration)
#> An S3 object representing a configuration.
#>
#> Number of points: 126.
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_exponential"
medium_range_model <- "square_exponential"
max_points <- 150
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_aic = TRUE,
                      saturation = saturation,
                      use_glmnet = FALSE)
#> $beta0
#> [1] -6.100179
#>
#> $alpha
#>           [,1]
#> [1,] 0.5072545
#>
```

```
#> $gamma
#>          [,1]
#> [1,] 2.710841
#> 
#> $beta
#>     
#> [1,]
print(summary(fit))
#>             coefficients        se     CI95_lo    CI95_hi Ztest       Pval
#> log_lambda1   -6.1001788 3.1463772 -12.2669647 0.06660715       0.05252636
#> alpha_1_1      0.5072545 0.6019795  -0.6726037 1.68711269       0.39942753
#> gamma_1_1      2.7108407 1.1921995   0.3741726 5.04750877     * 0.02297713
#>                   Zval
#> log_lambda1 -1.9387945
#> alpha_1_1    0.8426441
#> gamma_1_1    2.2738146
print(fit$coefficients)
#> $beta0
#> [1] -6.100179
#> 
#> $alpha
#>           [,1]
#> [1,] 0.5072545
#> 
#> $gamma
#>          [,1]
#> [1,] 2.710841
#> 
#> $beta
#>     
#> [1,]
#> 
#> $short_range
#>           [,1]
#> [1,] 0.6186606
#> 
#> $medium_range
#>          [,1]
#> [1,] 1.111072
#> 
#> $long_range
#>          [,1]
#> [1,] 7.515186
print(fit$aic)
#> [1] 623.2271
print(fit$bic)
#> [1] 636.5643
```

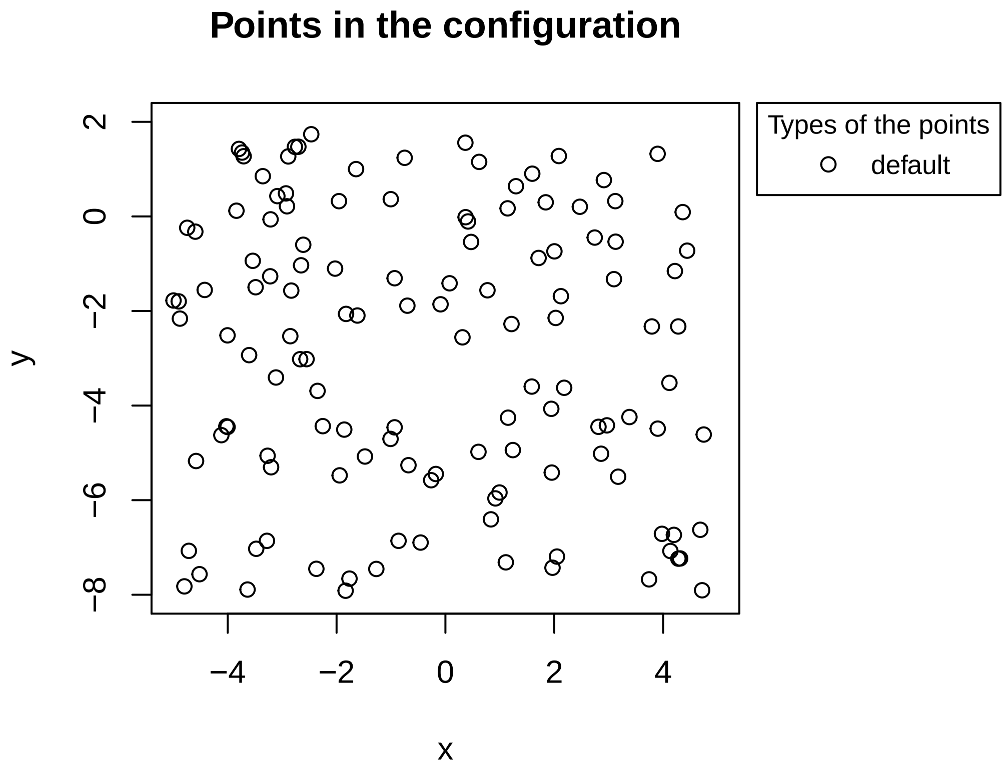
We may then plot the corresponding Papangelou conditional intensity.

```
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,
#                  max_points = max_points)
```

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))),
                       saturation = saturation,
                       steps = steps)
print(draw)
#> An S3 object representing a configuration.
#>
#> Number of points: 128.

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 saplings, and consider interaction radii in metres.

```
configuration <- Configuration(finpines$x, finpines$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_aic = TRUE,
                      saturation = saturation,
                      use_glmnet = FALSE)
#> $beta0
#> [1] -0.7517578
#>
#> $alpha
#>          [,1]
#> [1,] 0.858162
#>
```

```
#> $gamma
#>           [,1]
#> [1,] 0.3900442
#> 
#> $beta
#>     
#> [1,]
print(summary(fit))
#>              coefficients        se    CI95_lo    CI95_hi Ztest         Pval
#> log_lambda1   -0.7517578 0.3976302 -1.5310988 0.02758312       5.867840e-02
#> alpha_1_1      0.8581620 0.1520693  0.5601116 1.15621243   *** 1.668910e-08
#> gamma_1_1      0.3900442 0.3832373 -0.3610871 1.14117561       3.087913e-01
#>                  Zval
#> log_lambda1 -1.890595
#> alpha_1_1    5.643228
#> gamma_1_1    1.017762
print(fit$coefficients)
#> $beta0
#> [1] -0.7517578
#> 
#> $alpha
#>          [,1]
#> [1,] 0.858162
#> 
#> $gamma
#>           [,1]
#> [1,] 0.3900442
#> 
#> $beta
#>     
#> [1,]
#> 
#> $short_range
#>           [,1]
#> [1,] 0.2542557
#> 
#> $medium_range
#>         [,1]
#> [1,] 10.6263
#> 
#> $long_range
#>          [,1]
#> [1,] 21.17386
print(fit$aic)
#> [1] 583.4207
print(fit$bic)
#> [1] 596.7579
```

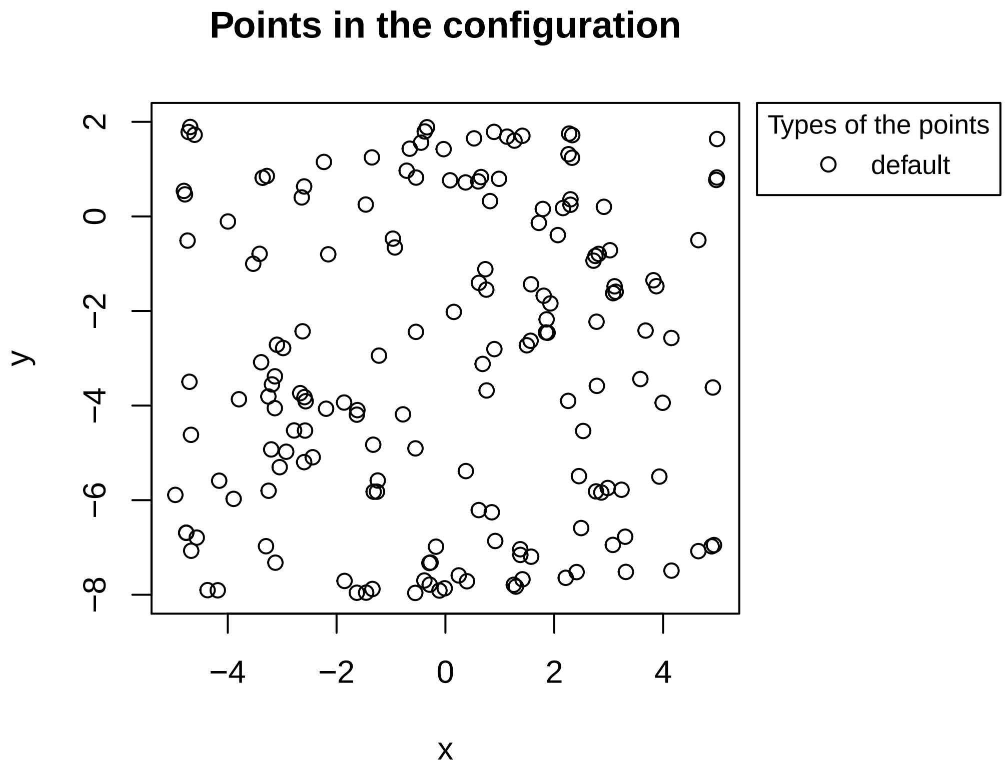
We may then plot the corresponding Papangelou conditional intensity.

```
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,
#                 lambda = beta0$beta0,
#                 beta = matrix(0, 1, 0),
#                 gamma = parameters$gamma,
#                 covariates = list(),
#                 short_range = parameters$short_range,
#                 medium_range = parameters$medium_range,
#                 long_range = parameters$long_range,
#                 saturation = saturation,
#                 max_points = max_points)
```

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)),
                       saturation = saturation,
                       mark_range = c(min(get_marks(configuration)), max(get_marks(configuration))),
                       steps = steps)
print(draw)
#> An S3 object representing a configuration.
#>
#> Number of points: 170.

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


Points in the configuration
Types of the points
○ default
y
2
0
−2
−4
−6
−8
x
−4
−2
0
2
4