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 1.63-0       (nickname: 'Space camouflage')
#> 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`.

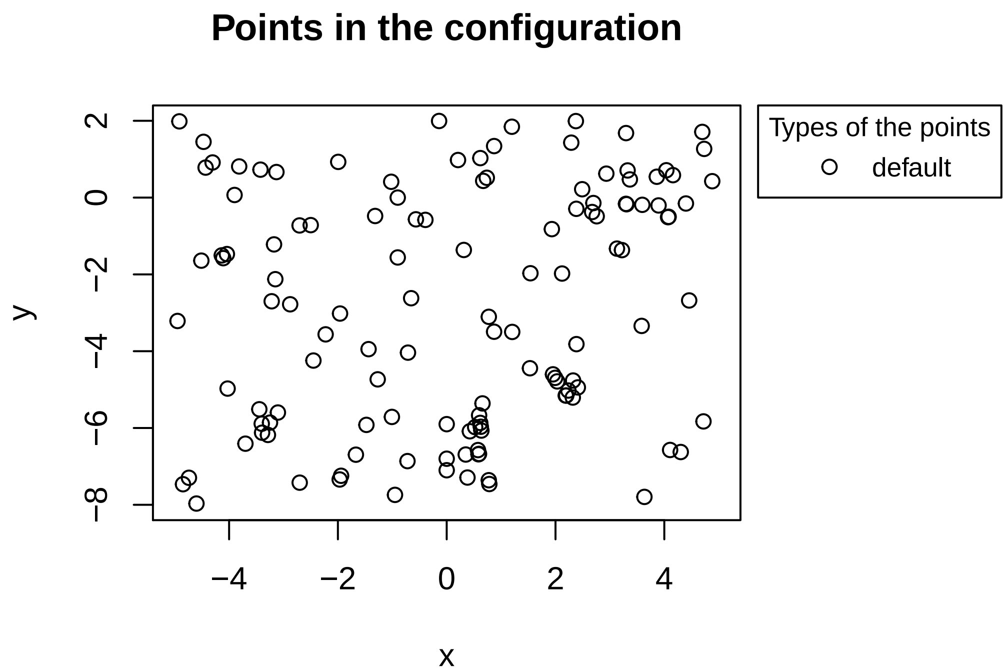
Taking marks into account

If marks are provided, the interaction radii are proportional to the marks. We begin with that setting.

```
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)
```

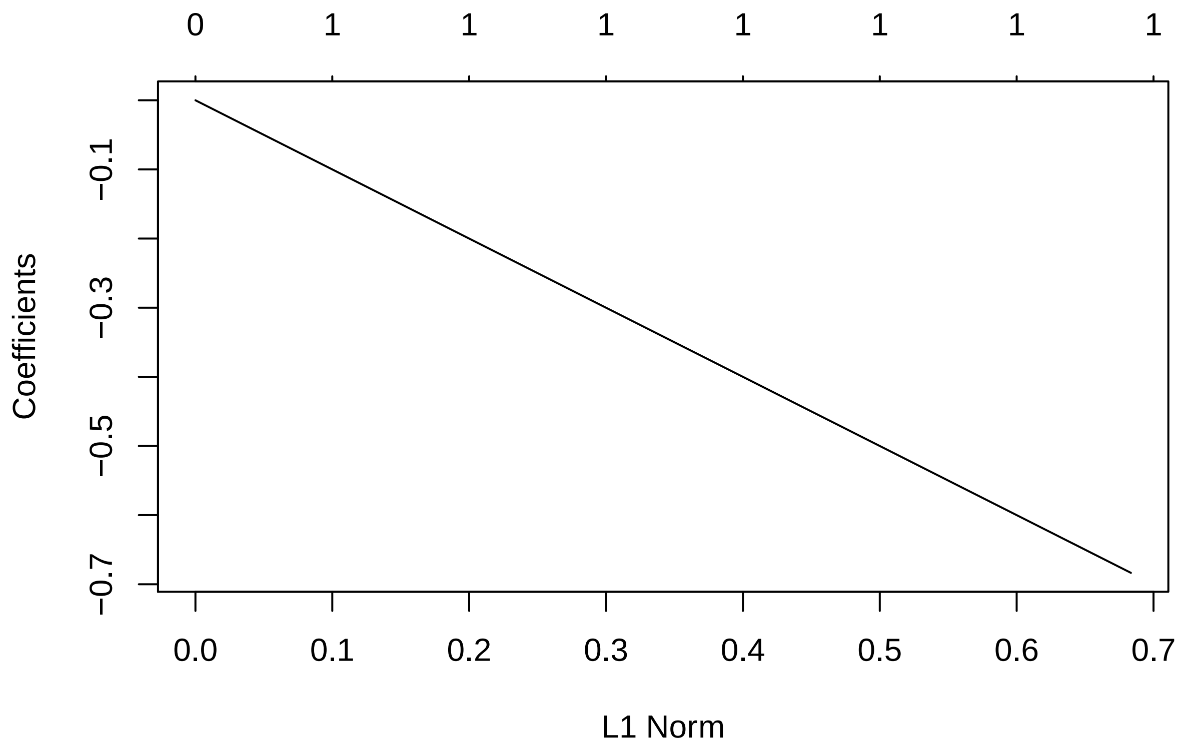


We provide a series of ranges for the interaction radii, and let the fitting function calibrate the model.

```
short_range <- c(0, 20)
medium_range <- c(0, 20)
long_range <- c(0, 20)
model <- "square_exponential"
medium_range_model <- "square_exponential"
```

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 = TRUE,
                      use_aic = TRUE,
                      saturation = 2)
#> (Intercept) log_lambda1    alpha_1_1    gamma_1_1
#>   0.0000000   1.6174061   -0.6834957    0.0000000
plot(fit$complete)
#> Warning in plotCoef(x$beta, lambda = x$lambda, df = x$df, dev = x$dev.ratio, : 1
#> or less nonzero coefficients; glmnet plot is not meaningful
```

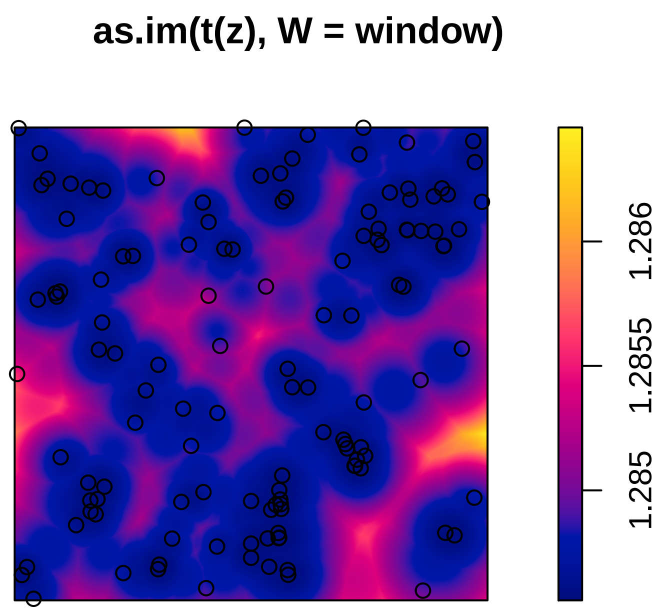



```
print(fit$coefficients)
#> (Intercept) log_lambda1   alpha_1_1   gamma_1_1
#>   0.0000000   1.6174061  -0.6834957   0.0000000
print(fit$best_short)
#>         [,1]
#> [1,] 15.2557
print(fit$best_medium)
#>          [,1]
#> [1,] 26.50541
print(fit$best_long)
#>          [,1]
#> [1,] 35.95118
print(fit$aic)
#> [1] -240.8092
print(fit$bic)
#> [1] -236.3699
```

We may then plot the corresponding Papangelou conditional intensity.

```
parameters <- get_parameters_from_fit(fit)
lambda <- parameters$lambda
alpha <- parameters$alpha
gamma <- parameters$gamma
plot_papangelou(window = window,
                configuration = configuration,
                type = 1,
                mark = mean(get_marks(configuration)),
```

```
                 model = model,
                 medium_range_model = medium_range_model,
                 alpha = alpha,
                 lambda = lambda,
                 beta = matrix(0, 1, 0),
                 gamma = gamma,
                 covariates = list(),
                 short_range = fit$best_short,
                 medium_range = fit$best_medium,
                 long_range = fit$best_long,
                 saturation = 2)
```

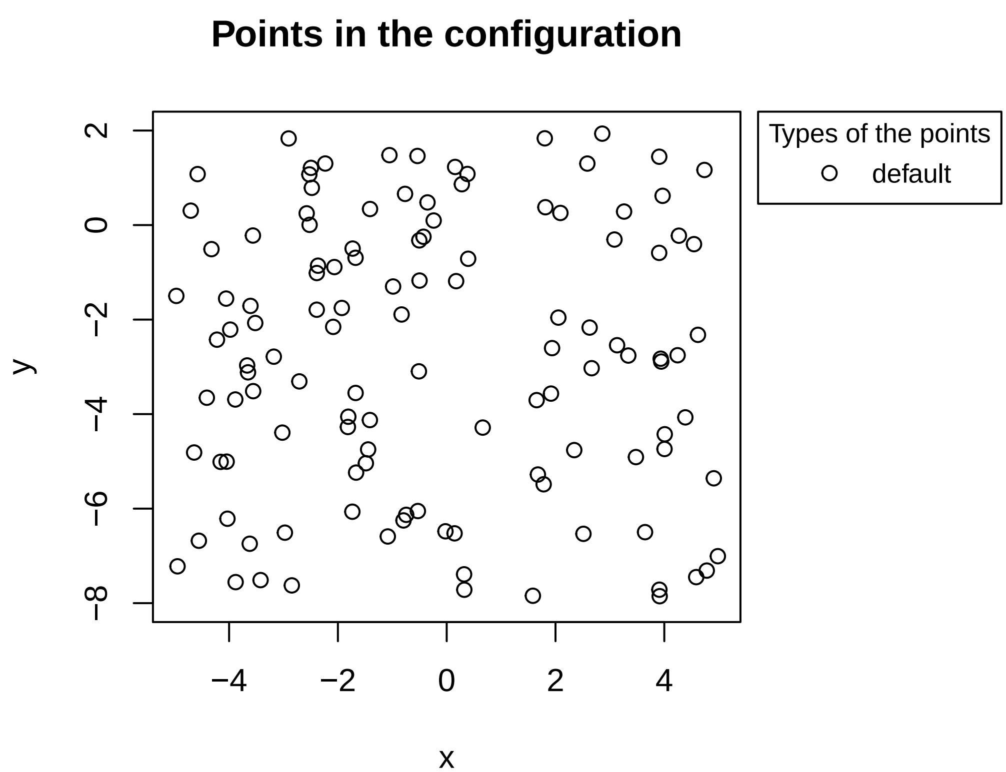


It is also possible to draw from the model.

```
parameters <- get_parameters_from_fit(fit)
lambda <- parameters$lambda
alpha <- parameters$alpha
gamma <- parameters$gamma
draw <- ppjsdm::rgibbs(window = window,
                       alpha = alpha,
                       lambda = lambda,
                       gamma = gamma,
                       model = model,
                       medium_range_model = medium_range_model,
                       short_range = fit$best_short,
                       medium_range = fit$best_medium,
                       long_range = fit$best_long,
```

```
                           types = levels(types(configuration)),
                           mark_range = c(min(get_marks(configuration)), max(get_marks(configuration))),
                           steps = 10000000)
print(draw)
#> An S3 object representing a configuration.
#>
#> Number of points: 119.

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



Interaction radii not proportional to marks

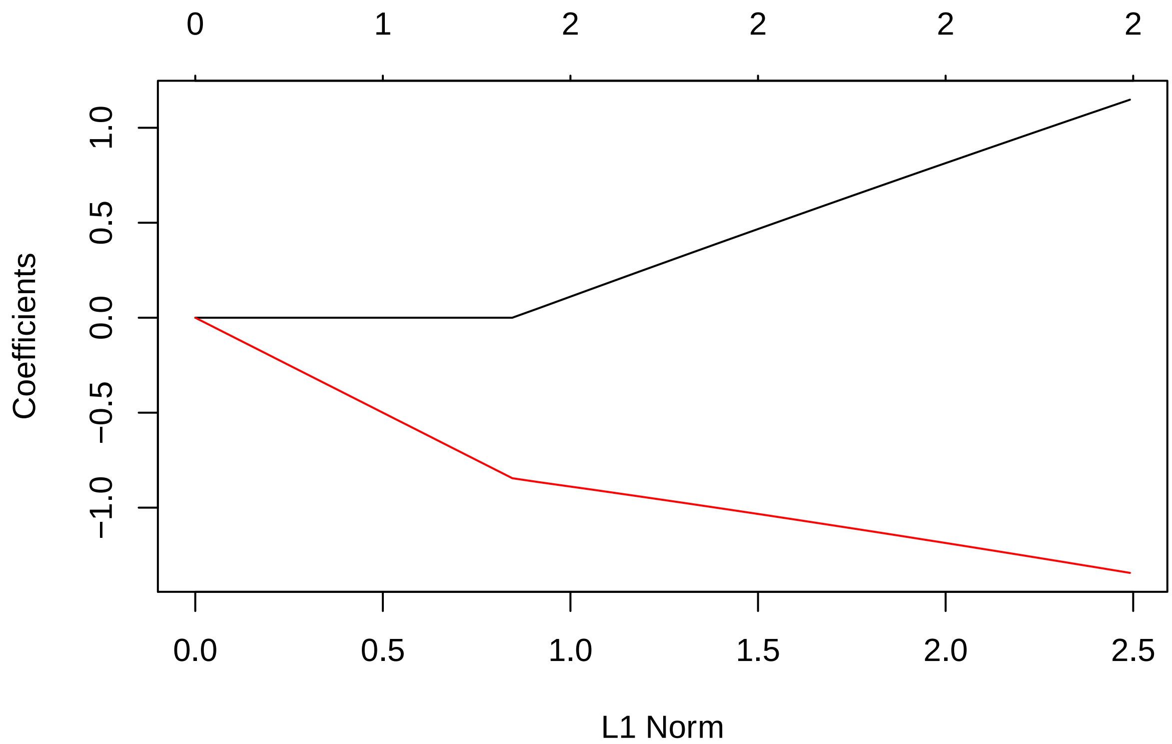
In this section, we do not account for marks.

```
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,
```

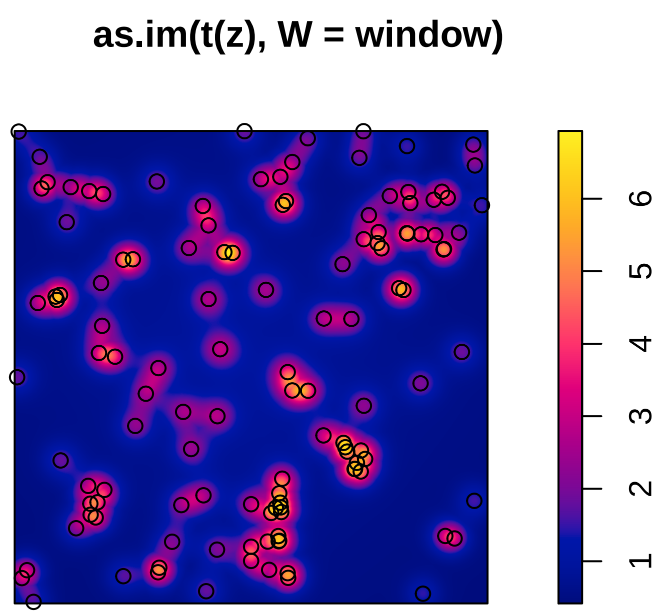
```
                        long_range = long_range,
                        use_glmnet = TRUE,
                        use_aic = TRUE,
                        saturation = 2)
#> (Intercept) log_lambda1    alpha_1_1    gamma_1_1
#>    0.000000    1.617406     1.147943    -1.343462
plot(fit$complete)
```




```
print(fit$coefficients)
#> (Intercept) log_lambda1    alpha_1_1    gamma_1_1
#>    0.000000    1.617406     1.147943    -1.343462
print(fit$best_short)
#>           [,1]
#> [1,] 0.3419659
print(fit$best_medium)
#>          [,1]
#> [1,] 11.41415
print(fit$best_long)
#>          [,1]
#> [1,] 25.13677
print(fit$aic)
#> [1] -274.2492
print(fit$bic)
#> [1] -265.3769
```

We may then plot the corresponding Papangelou conditional intensity.

```
parameters <- get_parameters_from_fit(fit)
lambda <- parameters$lambda
alpha <- parameters$alpha
gamma <- parameters$gamma
plot_papangelou(window = window,
                configuration = configuration,
                type = 1,
                mark = mean(get_marks(configuration)),
                model = model,
                medium_range_model = medium_range_model,
                alpha = alpha,
                lambda = lambda,
                beta = matrix(0, 1, 0),
                gamma = gamma,
                covariates = list(),
                short_range = fit$best_short,
                medium_range = fit$best_medium,
                long_range = fit$best_long,
                saturation = 2)
```



And as previously, we draw from the model.

```
parameters <- get_parameters_from_fit(fit)
lambda <- parameters$lambda
alpha <- parameters$alpha
gamma <- parameters$gamma
```

```r
draw <- ppjsdm::rgibbs(window = window,
                       alpha = alpha,
                       lambda = lambda,
                       gamma = gamma,
                       model = model,
                       medium_range_model = medium_range_model,
                       short_range = fit$best_short,
                       medium_range = fit$best_medium,
                       long_range = fit$best_long,
                       types = levels(types(configuration)),
                       mark_range = c(min(get_marks(configuration)), max(get_marks(configuration))),
                       steps = 10000000)
print(draw)
#> An S3 object representing a configuration.
#>
#> Number of points: 176.

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