Lansing dataset

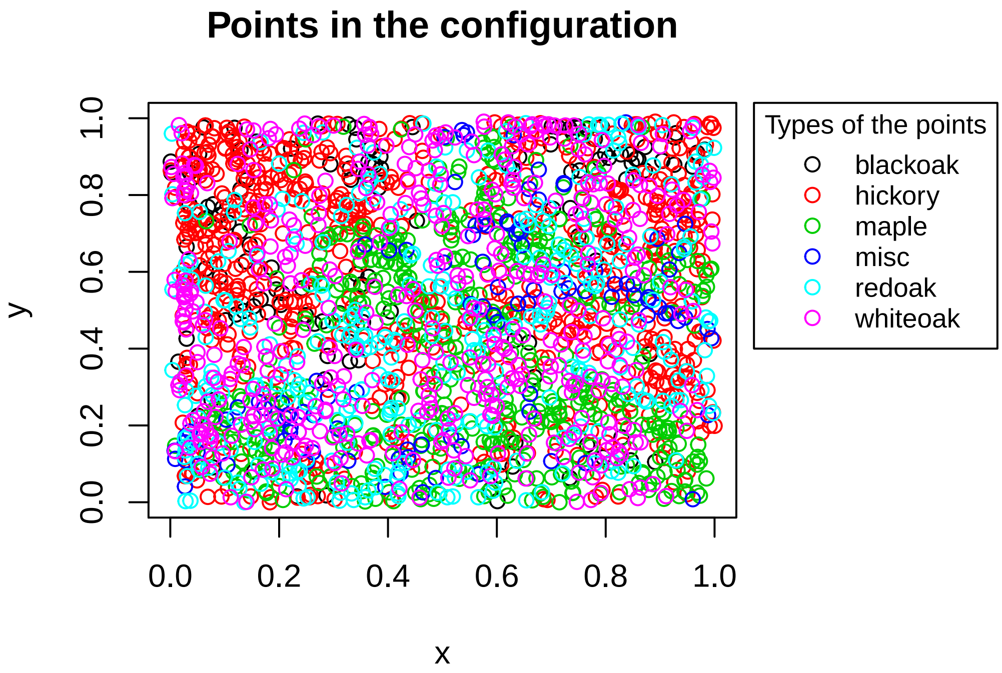
```
library(ppjsdm)
library(spatstat)
#> Loading required package: spatstat.data
#> Loading required package: nlme
#> Loading required package: rpart
#>
#> spatstat 1.63-0       (nickname: 'Space camouflage')
#> For an introduction to spatstat, type 'beginner'
remove(list = ls())

set.seed(1)
```

This vignette explains how to use the `ppjsdm` package with the Lansing dataset from `spatstat`. We begin by loading the data with all species.

```
data(lansing)
configuration <- as.Configuration(lansing)
window <- Rectangle_window(c(0, 1), c(0, 1))
```

The point configuration is plotted below.

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

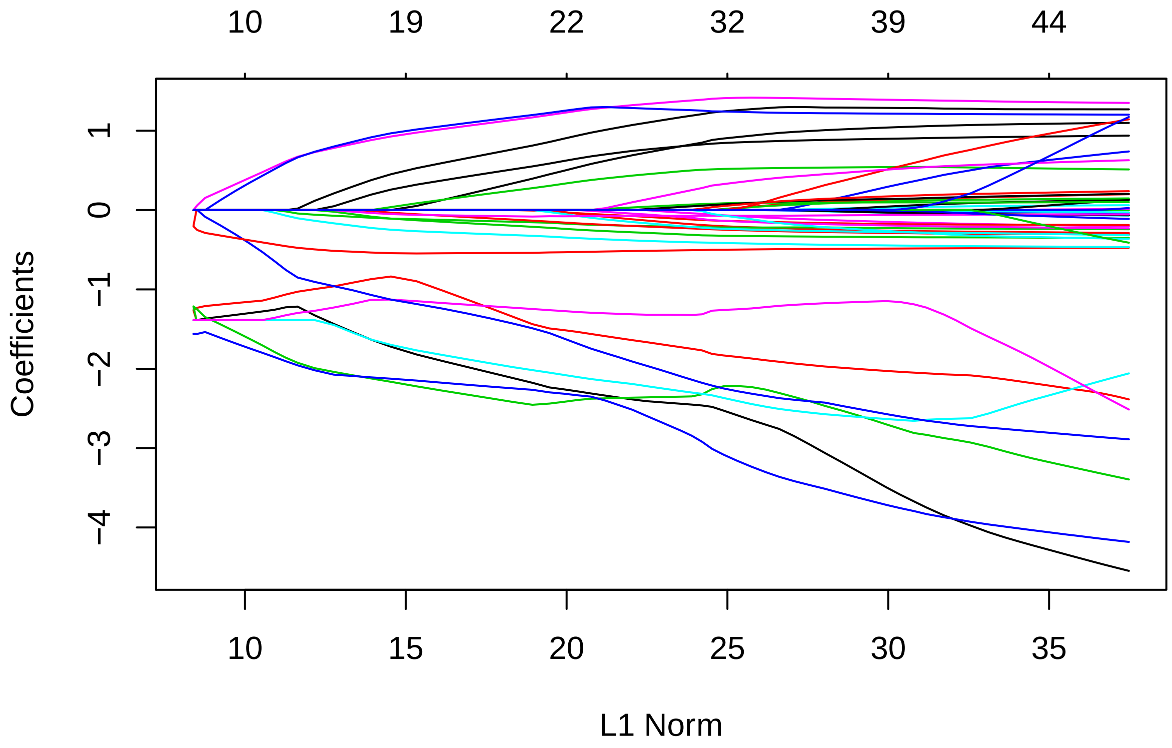


We fit the data with the Geyer model.

```
short_range <- c(0, 0.1)
medium_range <- c(0, 0.1)
long_range <- c(0, 0.1)
model <- "square_bump"
medium_range_model <- "square_exponential"
```

The matrix `radii` models interaction radii within a species, and between species.

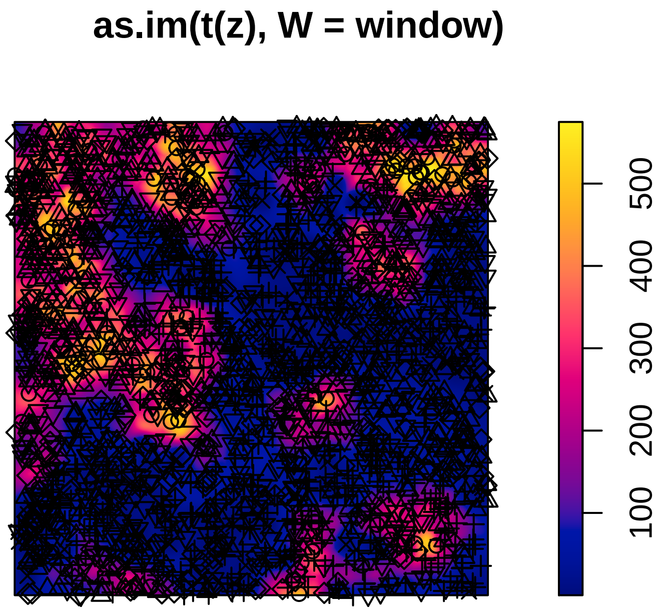
```
library(GA)
#> Loading required package: foreach
#> Loading required package: iterators
#> Registered S3 method overwritten by 'cli':
#>   method     from
#>   print.boxx spatstat
#> Package 'GA' version 3.2
#> Type 'citation("GA")' for citing this R package in publications.
#>
#> Attaching package: 'GA'
#> The following object is masked from 'package:utils':
#>
#>     de
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)
#>         (Intercept) shifted_log_lambda1 shifted_log_lambda2 shifted_log_lambda3
#>         0.000000000         1.951228461         5.698058134         4.404886250
#> shifted_log_lambda4 shifted_log_lambda5 shifted_log_lambda6           alpha_1_1
#>         2.126328707         4.956884463         5.401294264         1.269946516
#>           alpha_1_2           alpha_1_3           alpha_1_4           alpha_1_5
#>        -0.282293460        -0.238050263        -2.824272293        -0.062633135
#>           alpha_1_6           alpha_2_2           alpha_2_3           alpha_2_4
#>        -0.193683296         1.090959696        -0.477229588         0.000000000
#>           alpha_2_5           alpha_2_6           alpha_3_3           alpha_3_4
#>        -0.029437068        -0.461679620         1.357647230        -0.059112559
#>           alpha_3_5           alpha_3_6           alpha_4_4           alpha_4_5
#>        -0.186497441        -0.348321440         1.204809229        -0.022587330
#>           alpha_4_6           alpha_5_5           alpha_5_6           alpha_6_6
#>        -0.222835173         0.929707515        -0.302092571         0.521135149
#>           gamma_1_1           gamma_1_2           gamma_1_3           gamma_1_4
#>         0.654463290         0.057139477         0.008525546         0.115030672
#>           gamma_1_5           gamma_1_6           gamma_2_2           gamma_2_3
#>         0.183628321         0.140219636         0.000000000        -0.304917935
#>           gamma_2_4           gamma_2_5           gamma_2_6           gamma_3_3
#>        -0.055647074         0.074849670         0.000000000         1.001921642
#>           gamma_3_4           gamma_3_5           gamma_3_6           gamma_4_4
#>         0.098435822        -0.086637654        -0.348059414         0.605807220
#>           gamma_4_5           gamma_4_6           gamma_5_5           gamma_5_6
#>         0.189267056         0.224372856         0.000000000        -0.249581615
#>           gamma_6_6
#>         0.785722135
plot(fit$complete)
```



```
print(fit$coefficients)
#>         (Intercept) shifted_log_lambda1 shifted_log_lambda2 shifted_log_lambda3
#>         0.000000000         1.951228461         5.698058134         4.404886250
#> shifted_log_lambda4 shifted_log_lambda5 shifted_log_lambda6           alpha_1_1
#>         2.126328707         4.956884463         5.401294264         1.269946516
#>           alpha_1_2           alpha_1_3           alpha_1_4           alpha_1_5
#>        -0.282293460        -0.238050263        -2.824272293        -0.062633135
#>           alpha_1_6           alpha_2_2           alpha_2_3           alpha_2_4
#>        -0.193683296         1.090959696        -0.477229588         0.000000000
#>           alpha_2_5           alpha_2_6           alpha_3_3           alpha_3_4
#>        -0.029437068        -0.461679620         1.357647230        -0.059112559
#>           alpha_3_5           alpha_3_6           alpha_4_4           alpha_4_5
#>        -0.186497441        -0.348321440         1.204809229        -0.022587330
#>           alpha_4_6           alpha_5_5           alpha_5_6           alpha_6_6
#>        -0.222835173         0.929707515        -0.302092571         0.521135149
#>           gamma_1_1           gamma_1_2           gamma_1_3           gamma_1_4
#>         0.654463290         0.057139477         0.008525546         0.115030672
#>           gamma_1_5           gamma_1_6           gamma_2_2           gamma_2_3
#>         0.183628321         0.140219636         0.000000000        -0.304917935
#>           gamma_2_4           gamma_2_5           gamma_2_6           gamma_3_3
#>        -0.055647074         0.074849670         0.000000000         1.001921642
#>           gamma_3_4           gamma_3_5           gamma_3_6           gamma_4_4
#>         0.098435822         -0.086637654         -0.348059414          0.605807220
#>           gamma_4_5            gamma_4_6            gamma_5_5            gamma_5_6
#>         0.189267056          0.224372856          0.000000000         -0.249581615
#>           gamma_6_6
#>         0.785722135
print(fit$best_short)
#>            [,1]       [,2]       [,3]       [,4]       [,5]       [,6]
#> [1,] 0.07438044 0.02074919 0.02074919 0.02074919 0.02074919 0.02074919
#> [2,] 0.02074919 0.05015042 0.02074919 0.02074919 0.02074919 0.02074919
#> [3,] 0.02074919 0.02074919 0.04098474 0.02074919 0.02074919 0.02074919
#> [4,] 0.02074919 0.02074919 0.02074919 0.04907959 0.02074919 0.02074919
#> [5,] 0.02074919 0.02074919 0.02074919 0.02074919 0.04214092 0.02074919
#> [6,] 0.02074919 0.02074919 0.02074919 0.02074919 0.02074919 0.03608095
print(fit$best_medium)
#>            [,1]       [,2]       [,3]       [,4]       [,5]       [,6]
#> [1,] 0.05557372 0.06441114 0.06441114 0.06441114 0.06441114 0.06441114
#> [2,] 0.06441114 0.04227391 0.06441114 0.06441114 0.06441114 0.06441114
#> [3,] 0.06441114 0.06441114 0.06181586 0.06441114 0.06441114 0.06441114
#> [4,] 0.06441114 0.06441114 0.06441114 0.05579612 0.06441114 0.06441114
#> [5,] 0.06441114 0.06441114 0.06441114 0.06441114 0.06670669 0.06441114
#> [6,] 0.06441114 0.06441114 0.06441114 0.06441114 0.06441114 0.03620342
print(fit$best_long)
#>           [,1]       [,2]       [,3]       [,4]      [,5]       [,6]
#> [1,] 0.1078154 0.10137172 0.10137172 0.10137172 0.1013717 0.10137172
#> [2,] 0.1013717 0.07480855 0.10137172 0.10137172 0.1013717 0.10137172
#> [3,] 0.1013717 0.10137172 0.09867876 0.10137172 0.1013717 0.10137172
#> [4,] 0.1013717 0.10137172 0.10137172 0.09012743 0.1013717 0.10137172
#> [5,] 0.1013717 0.10137172 0.10137172 0.10137172 0.1156353 0.10137172
#> [6,] 0.1013717 0.10137172 0.10137172 0.10137172 0.1013717 0.08114627
print(fit$aic)
#> [1] -5368.356
```

We may then plot the corresponding Papangelou conditional intensity.

```
parameters <- get_parameters_from_fit(fit)
lambda <- parameters$lambda
alpha <- parameters$alpha
gamma <- parameters$gamma
plot_papangelou(window = window,
                configuration = configuration,
                type = 1,
                model = model,
                medium_range_model = medium_range_model,
                alpha = alpha,
                lambda = lambda,
                beta = matrix(0, 6, 0),
                gamma = gamma,
                covariates = list(),
                short_range = fit$best_short,
                medium_range = fit$best_medium,
                long_range = fit$best_long,
                saturation = 2)
#> Warning: data contain duplicated points
```


as.im(t(z), W = window)
500
400
300
200
100