# Fithian et al. (2014) NSW

```
library(maptools)
#> Loading required package: sp
#> Checking rgeos availability: FALSE
#>      Note: when rgeos is not available, polygon geometry     computations in maptools depend on gpcl
#>      which has a restricted licence. It is disabled by default;
#>      to enable gpclib, type gpclibPermit()
library(ppjsdm)
library(raster)
library(sf)
#> Linking to GEOS 3.6.2, GDAL 2.2.3, PROJ 4.9.3
library(spatstat)
#> Loading required package: spatstat.data
#> Loading required package: nlme
#>
#> Attaching package: 'nlme'
#> The following object is masked from 'package:raster':
#>
#>     getData
#> Loading required package: rpart
#>
#> spatstat 1.63-0       (nickname: 'Space camouflage')
#> For an introduction to spatstat, type 'beginner'
#>
#> Attaching package: 'spatstat'
#> The following objects are masked from 'package:raster':
#>
#>     area, rotate, shift
remove(list = ls())

source("../R/get_nsw.R")

set.seed(1)
```

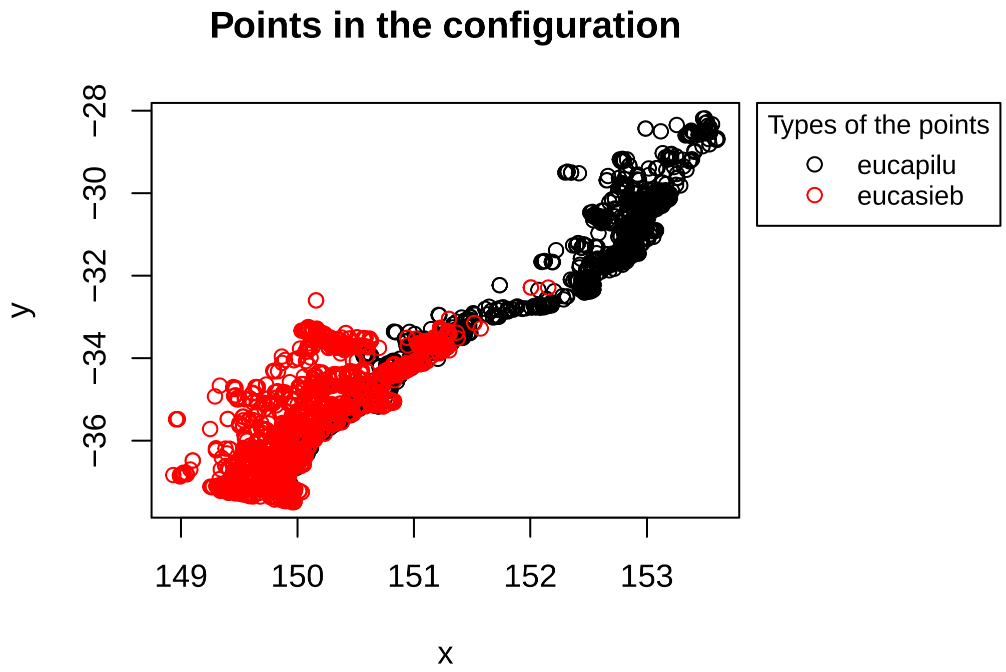
This vignette explains how to use the `ppjsdm` package with the NSW dataset from Fithian et al. (2014). We begin by loading the data with only the most prevalent species.

```
number_of_species <- 2 # Includes the most prevalent species from the plot

nsw <- get_nsw(prevalent = number_of_species)
configuration <- nsw$configuration
window <- nsw$window
covariates <- nsw$covariates
```

The point configuration is plotted below.

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

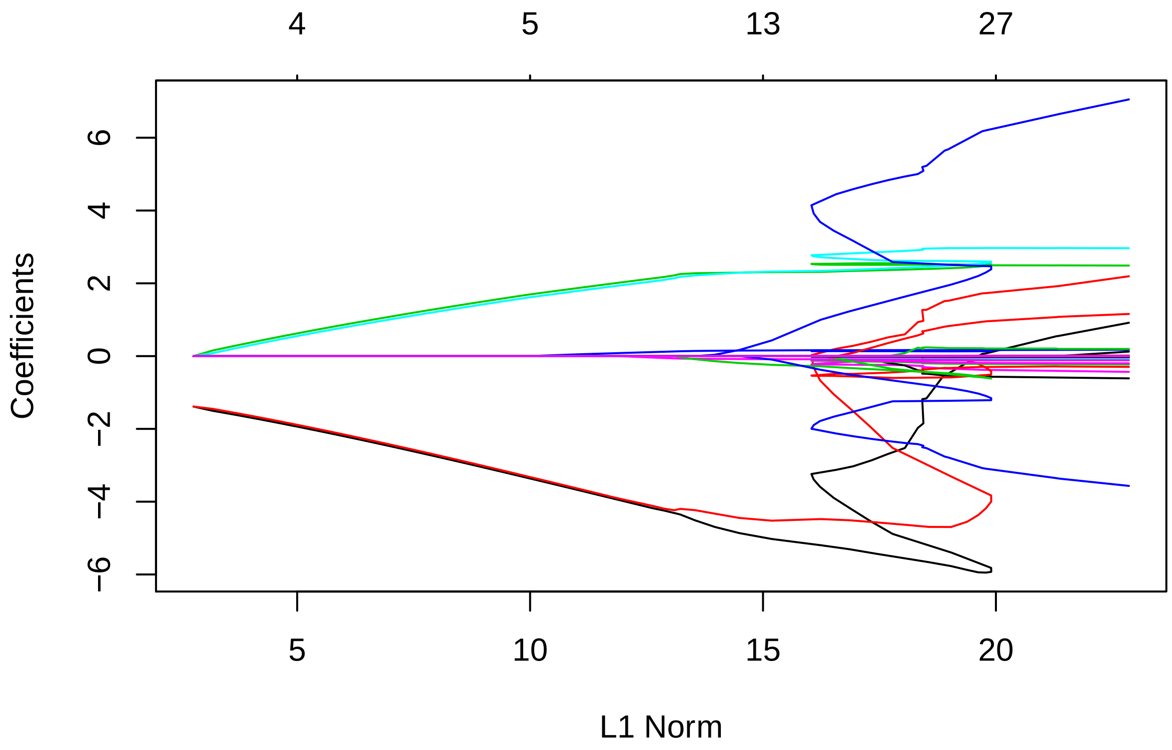


The matrix `radii` defined below models interaction radii within a species (on the diagonal), and between species (outside the diagonal).

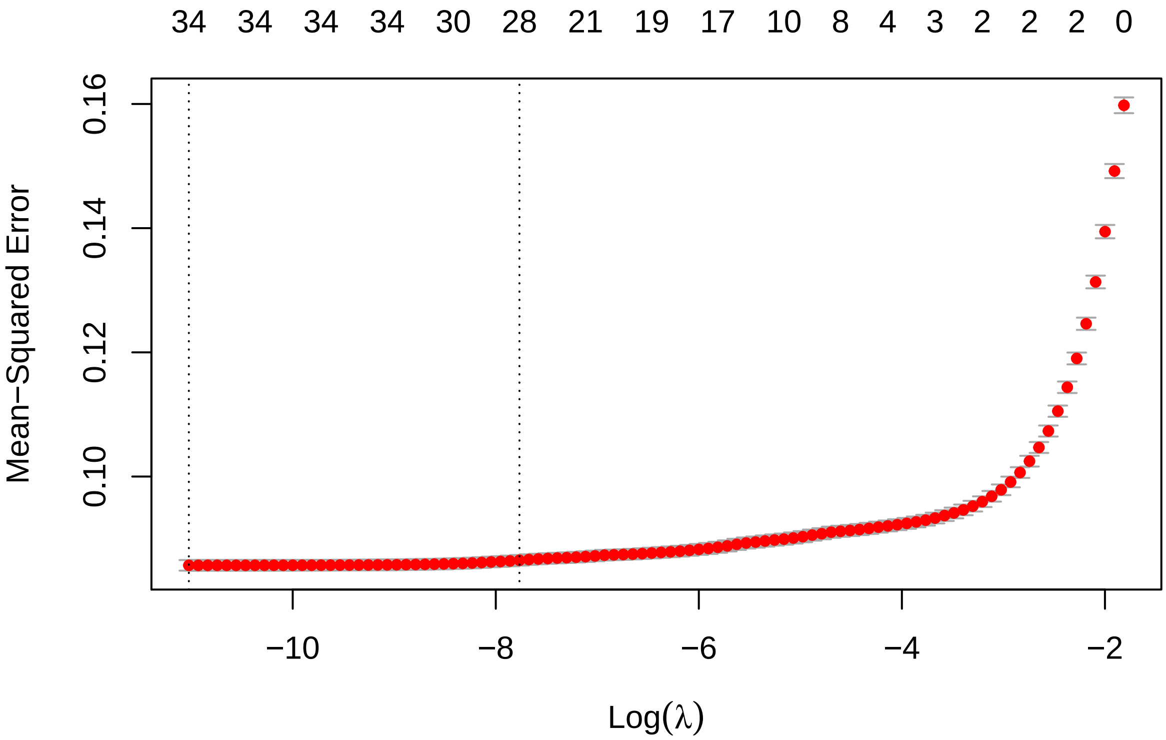
```
radii <- matrix(0.1, number_of_species, number_of_species)
```

Fitting the model to the dataset is then quite easy.

```
fit <- ppjsdm::gibbsm(configuration,
                      window = window,
                      covariates = covariates,
                      model = "Geyer",
                      radius = radii,
                      use_glmnet = TRUE)
#> 36 x 1 sparse Matrix of class "dgCMatrix"
#>                                   1
#> (Intercept)            .
#> log_lambda1            7.199701e+00
#> log_lambda2            8.494938e+00
#> alpha_1_1              2.488487e+00
#> alpha_1_2              1.759667e-01
#> alpha_2_2              2.965345e+00
#> unnamed_covariate1_1  -4.330249e-01
#> unnamed_covariate1_2   .
#> unnamed_covariate2_1  -6.119219e-01
#> unnamed_covariate2_2  -2.959237e-01
#> unnamed_covariate3_1   .
#> unnamed_covariate3_2  -2.269559e-01
#> unnamed_covariate4_1   1.882519e-03
#> unnamed_covariate4_2  -6.595800e-05
#> unnamed_covariate5_1   .
#> unnamed_covariate5_2  -1.982642e-01
#> unnamed_covariate6_1   .
#> unnamed_covariate6_2   .
#> unnamed_covariate7_1   1.245664e-01
#> unnamed_covariate7_2   1.157883e+00
#> unnamed_covariate8_1   1.944090e-01
#> unnamed_covariate8_2   7.053287e+00
#> unnamed_covariate9_1  -6.408781e-02
#> unnamed_covariate9_2  -1.146683e-01
#> unnamed_covariate10_1 -6.838757e-03
#> unnamed_covariate10_2 -4.301862e-03
#> unnamed_covariate11_1  .
#> unnamed_covariate11_2 -3.566867e+00
#> unnamed_covariate12_1 -1.658447e-02
#> unnamed_covariate12_2 -2.012602e-03
#> unnamed_covariate13_1 -2.747452e-02
#> unnamed_covariate13_2  1.490378e-02
#> unnamed_covariate14_1 -8.136612e-05
#> unnamed_covariate14_2 -1.912735e-04
#> unnamed_covariate15_1 -1.849664e-05
#> unnamed_covariate15_2 -6.555477e-05
plot(fit$complete)
```



```
plot(fit$cv)
```


34 34 34 34 30 28 21 19 17 10 8 4 3 2 2 2 0
Mean−Squared Error
0.10
0.12
0.14
0.16
−10
−8
−6
−4
−2
Log(λ)