Simulating scenarios of feature distribution

Bernardo Niebuhr

02 December, 2021

Contents

Introduction

There are a few big questions that underlie the assessment of the effects of anthropogenic infrastructure on wildlife. When performing environmental impact assessments, one aims not only to to find which factors affect wildlife and how strongly, but also (i) at which spatial scale there are effects and (ii) how these effects sum and interact when (as it is often the case) there are multiple infrastructures and vectors of landscape modification. The first question is also knows as the Zone of Influence (ZoI) problem. The second is generally tackled in the context of cumulative impact assessment.

In the main text of this manuscript, we argue that the density of infrastructure features might better represent the cumulative effect of multiple features in the landscape than considering only the distance to the feature nearest to a given location. This, however, might vary depending on how the number of features in the landscape, how these features are distributed in space, and what is the ZoI of each of those features.

Here we simulate some landscapes with point-type infrastructure spread following different patterns, calculate the distance to the nearest feature and the density of features, at multiple scales, and assess when and how these variables might represent different sources of spatial variation.

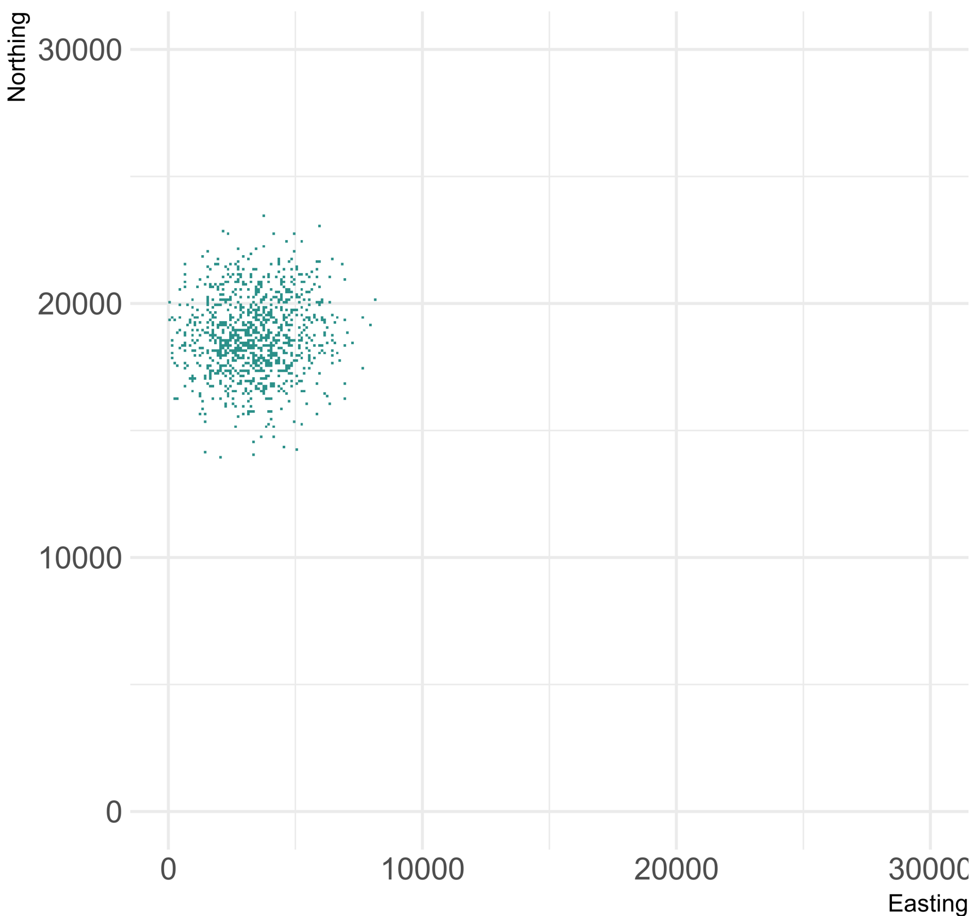
Simulate landscapes

First we simulate some landscape using point-type infrastructure as an example. They could represent the spatial location of houses, cabins, or wind turbines, for example. To do this we'll use a few functions designed within the package `oneimpact`.

```r
# Load packages
library(dplyr)
library(purrr)
library(ggplot2)

library(mobsim)
library(raster)
library(landscapetools)
library(sf)
library(spatstat)

library(oneimpact)

library(reshape2)
library(lsr)
```

We set a 30x30 km^2 landscape and can create simulate points following different spatial patterns. Here is an example landscape where the points are spread close to a single center (e.g. houses in a village).

```r
set.seed(1234)

# simulate a single patch
nfeat <- 1000 # number of features
ext <- 30000 # extension of the landscape
nc <- 1 # number of centers or patches
wd <- ext / 20 # width of the patch
pts <- set_points(
  n_features = 1000, centers = nc,
  width = wd, res = 100,
  extent_x = c(0, ext), extent_y = c(0, ext)
)

landscapetools::show_landscape(pts$rast, legend.position = "none")
```

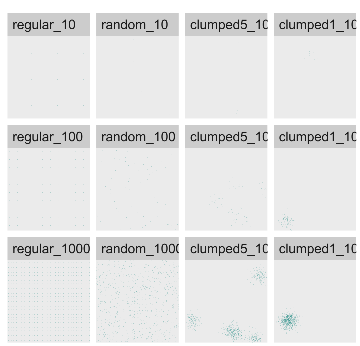


We can now simulate landscapes with different patterns. We start with 3 scenarios: regular, random, and clumped distribution of points. Each scenario is simulated with 10, 100, and 1000 points, so that we have 12 scenarios in total.

```r
set.seed(123)

# random, gradient, single center, multiple centers
methods <- c("regular", "random", "mobsim")
name <- c("regular", "random", "clumped5", "clumped1")
nfeat <- c(10, 100, 1000) # number of features
res <- 100 # resolution
```

```r
ext <- 30000 # extent of the landscape
nc <- c(5, 1) # number of centers for clumped
wd <- c(0.05) * ext # width of the "patches"

# parameters
parms_df1 <- expand.grid(
  method = methods, n_features = nfeat,
  centers = nc[1], width = wd
)
parms_df2 <- expand.grid(
  method = methods[3], n_features = nfeat,
  centers = nc[2], width = wd
)
parms_df <- dplyr::bind_rows(parms_df1, parms_df2) %>%
  dplyr::arrange(n_features, method)
scenarios <- paste0(rep(name, 3), "_", rep(nfeat, each = 4))

# simulate points
pts <- parms_df %>% purrr::pmap(set_points,
  res = res,
  extent_x = c(0, ext),
  extent_y = c(0, ext)
)

landscapes <- purrr::map(pts, ~ .[[2]])
names(landscapes) <- scenarios

landscapetools::show_landscape(landscapes)
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.

## Warning: Removed 7188 rows containing missing values (geom_raster).
```




```r
# plot(landscapes, col = "black", nc = 4, legend = F)
# rasterVis::levelplot(stack(landscapes), layout = c(4,3), names.attr = scenarios,
#                      par.settings = GrTheme, colorkey = FALSE)
```

We also store some variables related the distance between points for each of these scenarios. These variables are: (i) the average distance between points, the average nearest neighbor distance, and the average isolation.

```r
# isolation to random points
isolation <- function(x, n_rand = 100, ext = c(0, 1), lonlat = FALSE) {
  # create random points
  rand <- data.frame(x = runif(n_rand, ext[1], ext[2]), y = runif(n_rand, ext[1], ext[2]))
  # calc dist
  dists <- pointDistance(x, rand, lonlat = lonlat)
```

```r
  # min dist
  apply(dists, 2, min)
}

# mean isolation
mean_isolation <- function(x, n_rand = 100, ext = c(0, 1), lonlat = FALSE) {
  mean(isolation(x, n_rand = n_rand, ext = ext, lonlat = lonlat))
}

# points
pts_coords <- purrr::map(pts, first)
names(pts_coords) <- scenarios

# calculate distances
dist_scenarios <- data.frame(
  scenario = scenarios,
  mean_dist = map_dbl(pts_coords, function(x) mean(dist(x))),
  mean_nndist = map_dbl(pts_coords, function(x) mean(spatstat.geom::nndist(x))),
  mean_isolation = map_dbl(pts_coords, mean_isolation, n_rand = 150, ext = c(0, ext))
)

dist_scenarios
```

```
##                    scenario mean_dist mean_nndist mean_isolation
## regular_10       regular_10 15510.736   9486.8330      3910.7559
## random_10         random_10 17450.119   5238.7641      5320.3333
## clumped5_10     clumped5_10 15197.147   1155.7217      8224.4716
## clumped1_10     clumped1_10  3029.684   1132.5368     11103.3612
## regular_100     regular_100 15717.795   3000.0000      1182.9837
## random_100       random_100 15252.246   1597.3085      1585.0777
## clumped5_100   clumped5_100  9454.507    668.2567      5048.0218
## clumped1_100   clumped1_100  2454.158    319.6537     13629.1097
## regular_1000   regular_1000 15590.481    948.6833       371.6643
## random_1000     random_1000 15765.710    486.1710       479.7201
## clumped5_1000 clumped5_1000 14179.990    208.8593      4900.2276
## clumped1_1000 clumped1_1000  2662.354    115.1179     10947.7893
```