Appendix

Eric Marcon

Florence Puech

January 30, 2023

Abstract

Code to reproduce the figures of the main text.

1 Theoretical example: homogeneous controls

Analyses rely on the *dbmss* (Marcon et al., 2015) package for R (R Core Team, 2022).

This example completes that of the main text, detailed in section 2.

1.1 Dataset simulation

We build a point pattern made of cases (the points of interest) and controls (the background distribution of points).

Cases are a Matérn (Matérn, 1960) point pattern with κ (expected) clusters of μ (expected) points in a circle of radius *scale*. Controls are a Poisson point pattern (i.e. complete spatial randomness) of λ (expected) points.

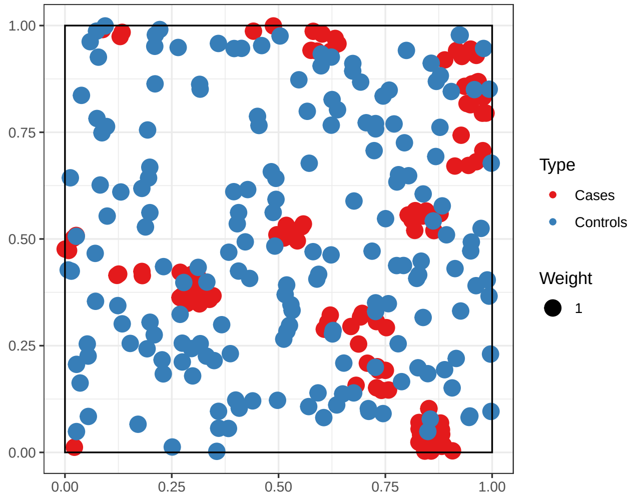
```r
library(dplyr)
library(dbmss)
# Simulation of cases (clusters)
rMatClust(kappa = 10, scale = 0.05, mu = 10) %>%
    as.wmppp -> CASES
CASES$marks$PointType <- "Cases"
# Number of points
CASES$n
```

```
## [1] 113
```

```r
# Simulation of controls (random distribution)
rpoispp(lambda = 200) %>%
    as.wmppp -> CONTROLS
CONTROLS$marks$PointType <- "Controls"
# Number of points
CONTROLS$n
```

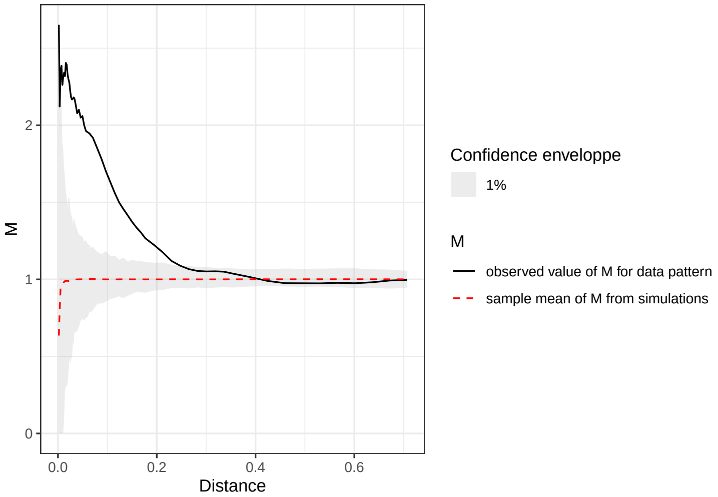
```
## [1] 185
```

```
# Mixed patterns (cases and controls)
ALL <- superimpose(CASES, CONTROLS)
autoplot(ALL)
```



1.2 Calculate and plot M Cases

```
# Fix the number of simulations and the level of
# risk
NumberOfSimulations <- 1000
Alpha <- 0.01
# Calculate and plot M Cases
ALL %>%
    MEnvelope(ReferenceType = "Cases", SimulationType = "RandomLocation",
        NumberOfSimulations = NumberOfSimulations,
        Alpha = Alpha, Global = TRUE) -> M_env_cases
autoplot(M_env_cases)
```



The plot shows a clear relative concentration of cases.

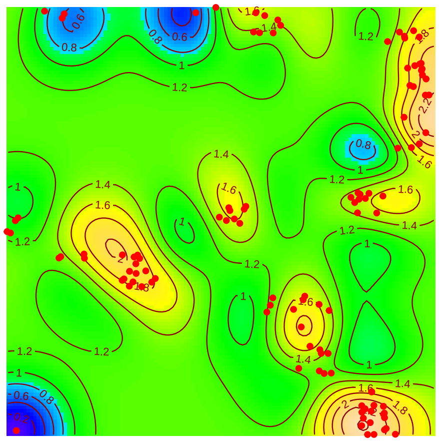
1.3 Map *M* results

To plot the individual values of M around each case, a distance must be chosen. Then, the function must be computed at this distance with individual values. Finally, a kriged weighted, marked, planar point patterns (`kwmppp`) object is produced and plotted.

```r
# Choose the distance to plot
Distance <- 0.1
# Calculate the M values to plot
ALL %>%
    Mhat(r = c(0, Distance), ReferenceType = "Cases",
        NeighborType = "Cases", Individual = TRUE) ->
    M_TheoEx
# Map resolution
resolution <- 512
# Create a kriged weighted marked planar point
# pattern (kwmppp)
M_TheoEx_map <- kwmppp(ALL, fvind = M_TheoEx, ReferenceType = "Cases",
    distance = Distance)
```

```
## [using ordinary kriging]
```

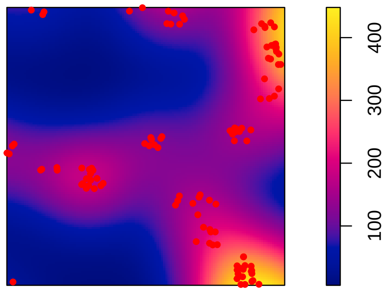
```r
# Plot the point pattern with values of
# M(Distance)
plot(M_TheoEx_map)
# Add the cases to the map
points(ALL[ALL$marks$PointType == "Cases"], pch = 20,
    col = "red")
```



1.4 Compare with the density of cases

The density of cases is plotted. High densities are similar to high relative concentrations in this example because the control points are homogeneously distributed.

```
plot(density(CASES), main = "")
points(ALL[ALL$marks$PointType == "Cases"], pch = 20,
    col = "red")
```



2 Theoretical example: non-homogeneuous controls

This section allows reproducing the theoretical example presented in the main text.

2.1 Dataset simulation

We build a point pattern made of cases (the points of interest) and controls (the background distribution of points).

Cases are a Matérn (Matérn, 1960) point pattern with κ (expected) clusters of μ (expected) points in a circle of radius *scale*. Controls are a Poisson point pattern whose density λ decreases exponentially along the y-axis (we will call "north" the higher y values).

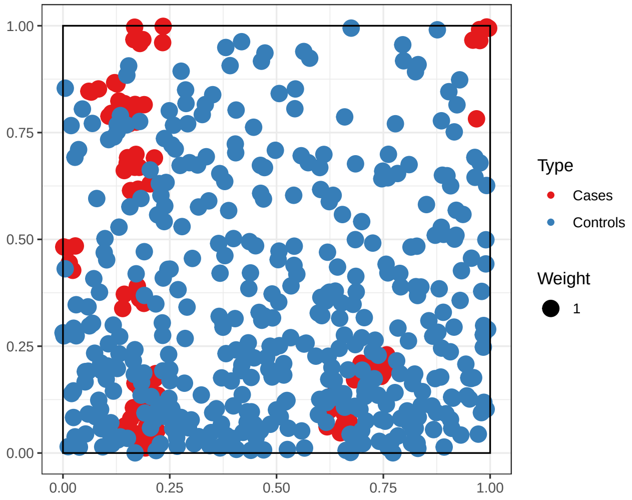
```
library(dbmss)
# Simulation of cases (clusters)
rMatClust(kappa = 10, scale = 0.05, mu = 10) %>%
    as.wmppp -> CASES
CASES$marks$PointType <- "Cases"
# Number of points
CASES$n
```

```
## [1] 94
```

```
# Simulation of controls (random distribution)
rpoispp(function(x, y) {
    1000 * exp(-2 * y)
}) %>%
    as.wmppp -> CONTROLS
CONTROLS$marks$PointType <- "Controls"
# Number of points
CONTROLS$n
```

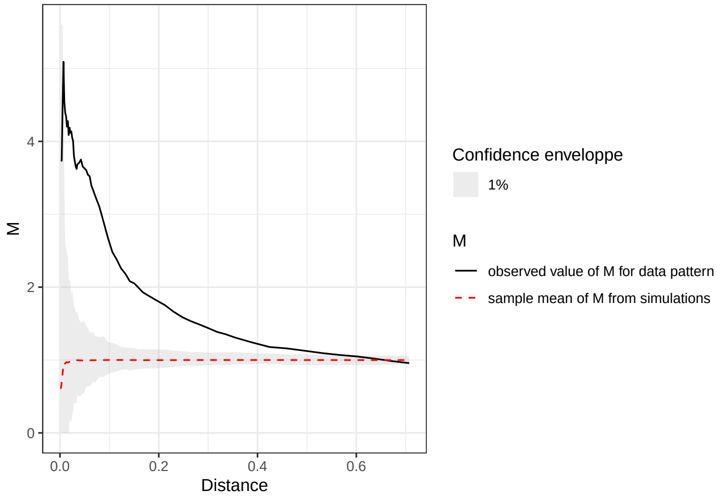
```
## [1] 428
```

```
# Mixed patterns (cases and controls)
ALL <- superimpose(CASES, CONTROLS)
autoplot(ALL)
```



2.2 Calculate and plot M Cases

```
# Fix the number of simulations and the level of
# risk
NumberOfSimulations <- 1000
Alpha <- 0.01
# Calculate and plot M Cases
ALL %>%
    MEnvelope(ReferenceType = "Cases", SimulationType = "RandomLocation",
        NumberOfSimulations = NumberOfSimulations,
        Alpha = Alpha, Global = TRUE) -> M_env_cases
autoplot(M_env_cases)
```



The plot shows a clear relative concentration of cases.

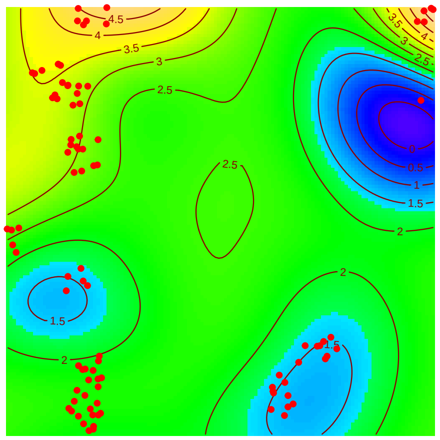
2.3 Map M results

To plot the individual values of M around each case, a distance must be chosen. Then, the function must be computed at this distance with individual values. Finally, a kriged weighted, marked, planar point patterns (`kwmppp`) object is produced and plotted.

```
# Choose the distance to plot
Distance <- 0.1
# Calculate the M values to plot
ALL %>%
    Mhat(r = c(0, Distance), ReferenceType = "Cases",
        NeighborType = "Cases", Individual = TRUE) ->
    M_TheoEx
# Map resolution
resolution <- 512
# Create a kriged weighted marked planar point
# pattern (kwmppp)
M_TheoEx_map <- kwmppp(ALL, fvind = M_TheoEx, ReferenceType = "Cases",
    distance = Distance)
```

```
## [using ordinary kriging]
```

```
# Plot the point pattern with values of
# M(Distance)
plot(M_TheoEx_map)
# Add the cases to the map
points(ALL[ALL$marks$PointType == "Cases"], pch = 20,
    col = "red")
```



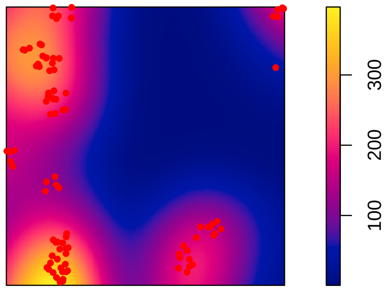
We can see that cases are concentrated everywhere (local M value above 1) because we chose a Matérn point pattern.

The areas with the higher relative concentration are located in the north of the map because the controls are less dense there. The southern-most cluster illustrates that the relative concentration of cases, although higher than 1, is clearly lower than that of the northern clusters, which have the same characteristics but are in a less-dense control neighborhood.

2.4 Compare with the density of cases

The density of cases is plotted. High densities are *not* similar to high relative concentrations in this example because the control points are not homogeneously distributed.

```
plot(density(CASES), main = "")
points(ALL[ALL$marks$PointType == "Cases"], pch = 20,
    col = "red")
```



3 Suzanne Lenglen Park

3.1 Data

Our data is extracted from "Paris open data" [1].

3.1.1 Data wrangling

Data are stored in `trees_2021.zip` which contains two GeoJSON files:

- trees_2021 stores all trees of the city of Paris in 2021.
- trees_logged contains all trees logged after 2021.

They must be read. Data are projected into the Lambert 93 datum so that coordinates are in meters.

```
unzip("data/trees_2021.zip", exdir = "data")
library("sf")
read_sf("data/trees_2021.geojson") %>%
    st_transform(crs = 2154) -> trees_all_raw
read_sf("data/trees_logged.geojson") %>%
    st_transform(crs = 2154) -> trees_logged_raw
```

[1] https://opendata.paris.fr.

All trees The first dataset contains all trees in Paris in 2021, including those to be cut.

Trees from the Suzanne Lenglen park are selected. Columns of interest are:

- `ID`: a numeric unique identifier for each tree.
- `Species_name`: the scientific name of the tree species, i.e. Genus species.
- `Status`: Alive.
- `Genus`.
- `Species`.
- `French_species_name`: vernacular name.
- `Circumference`: in cm.

```
library("dplyr")
trees_all_raw %>%
    # Filter Suzanne Lenglen park
filter(adresse == "PARC OMNISPORT SUZANNE LENGLEN / 7 BOULEVARD DES FRERES VOISIN") %>%
    # Create a field with the species name
mutate(Species_name = as.factor(paste(genre, espece))) %>%
    # Create a field with the status
mutate(Status = "Alive") %>%
    # Genus and Species fields
mutate(Genus = as.factor(genre)) %>%
    mutate(Species = as.factor(espece)) %>%
    # Rename and finally select columns
rename(ID = idbase, French_species_name = libellefrancais,
    Circumference = circonferenceencm) %>%
    select(ID, Species_name, Status, Genus, Species,
        French_species_name, Circumference) -> trees_all
# Number of trees
trees_all %>%
    nrow()
```

```
## [1] 1472
```

We have 1472 trees in the park.

Logged trees Logged trees are in the second dataset.

Their status is “Logged”. An extra field, `Logging_reason` contains the motivation to cut them off (in French). `Circumference` is absent.

```
# Tree description
trees_logged_raw %>%
    # Filter Suzanne Lenglen park
filter(adresse == "PARC OMNISPORT SUZANNE LENGLEN / 7 BOULEVARD DES FRERES VOISIN") %>%
    # Exclude unidentified trees
filter(!is.na(especearbreprecedent), !is.na(libellefrancaisarbreprecedent),
    !is.na(genrearbreprecedent)) %>%
    filter(libellefrancaisarbreprecedent != "Non spécifié") %>%
    filter(especearbreprecedent != "n. sp.") %>%
    # Create a field with the species name
mutate(Species_name = as.factor(paste(genrearbreprecedent,
    especearbreprecedent))) %>%
    # Create a numeric ID
mutate(ID = as.integer(idbase)) %>%
    # Create a field with the status
mutate(Status = "Logged") %>%
    # Genus and Species fields
mutate(Genus = as.factor(genrearbreprecedent)) %>%
    mutate(Species = as.factor(especearbreprecedent)) %>%
    # Reason for logging (in French)
```

```
mutate(Logging_reason = motifabattagearbreprecedent) %>%
    # Rename and finally select columns
rename(French_species_name = libellefrancaisarbreprecedent) %>%
    select(ID, Species_name, Status, Genus, Species,
        Logging_reason, French_species_name) ->
    trees_logged
# Number of trees
trees_logged %>%
    nrow()
```

```
## [1] 48
```

48 among the 1472 trees of the park were logged.

Merge The two datasets are merged here.

The logged trees must be removed from the first one. `Circumference` is removed because it is missing from the logged trees dataset.

```
# All trees
trees_all %>%
    # Delete the logged trees
filter(!(ID %in% trees_logged$ID)) %>%
    # Delete the circumference that is absent in
    # trees_logged
mutate(Circumference = NULL) %>%
    # Bind the logged trees
bind_rows(trees_logged) -> trees_no_circumference
```

Circumferences of all trees, including logged ones, are in `tree_all` from where they can be recovered.

```
# Prepare a tibble with circumferences
trees_all %>%
    select(ID, Circumference) %>%
    # inner_join.sf refuses sf objects
st_set_geometry(NULL) -> Circumferences
# Add the Circumference of trees
trees_no_circumference %>%
    inner_join(Circumferences, by = "ID") -> trees
```

Shorter logging reasons Logging reasons can be:

- Decaying: the tree's condition is not healthy enough to keep it safely in a public park.
- infested: the tree is a maple affected by the (contagious) sooty bark disease, caused by the fungus *Cryptostroma corticale*.

```
library("stringr")
trees$Logging_reason[is.na(trees$Logging_reason)] <- ""
trees$Logging_reason %>%
    str_replace("Arbre.*", "Decaying") %>%
    str_replace("Foyer.*", "infested") -> trees$Logging_reason
```

Factors Several fields are converted to factors for efficiency.

```
trees$Logging_reason <- as.factor(trees$Logging_reason)
trees$Status <- as.factor(trees$Status)
trees$French_species_name <- as.factor(trees$French_species_name)
```

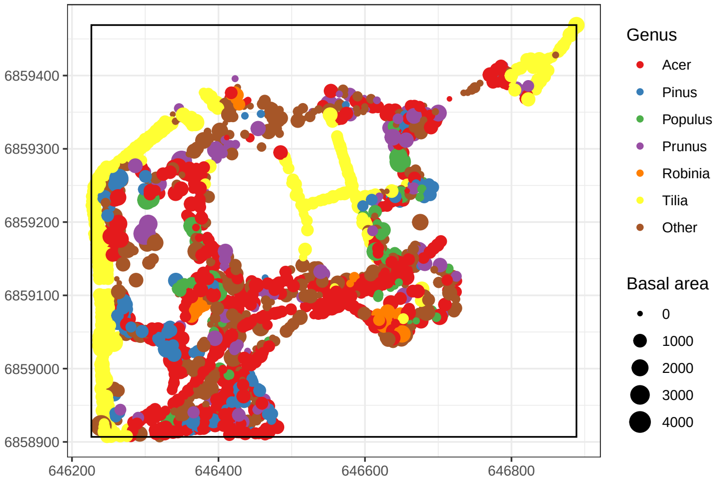
3.1.2 Point patterns

dbmms uses weighted, marked, planar point patterns (wmppp). A wmppp named `trees_infested` is built. Point marks are their basal area (as weight) and either their logging reason or their genus if they are alive.

```
library("dbmss")
trees %>%
    # Weight is the basal area
mutate(PointWeight = Circumference^2/4/pi) %>%
    mutate(PointType = ifelse(Logging_reason == "",
        as.character(Genus), as.character(Logging_reason))) %>%
    # Add X and Y
bind_cols(st_coordinates(trees)) %>%
    wmppp(window = as.owin(st_bbox(trees)), unitname = c("meter",
        "meters")) -> trees_infested
```

We also need a point pattern to describe the park before logging, as a reference.

```
trees_all %>%
    # Weight is the basal area
mutate(PointWeight = Circumference^2/4/pi) %>%
    # Genus is the point type
rename(PointType = Genus) %>%
    # Add X and Y
bind_cols(st_coordinates(trees_all)) %>%
    wmppp(window = as.owin(st_bbox(trees_all)), unitname = c("meter",
        "meters")) -> trees_2021
autoplot(trees_2021) + ggplot2::labs(size = "Basal area",
    color = "Genus")
```

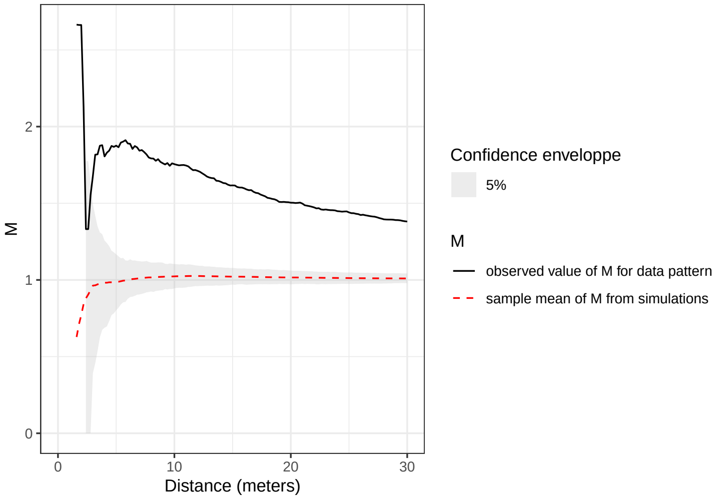


The map shows the tree genera. Maple (*Acer spp.*) are the most abundant trees in the park.

3.2 Spatial concentration of maple trees

The M statistic is computed to detect the spatial concentration of maple trees before logging.

```
Distance <- 15
NumberOfSimulations <- 1000
trees_2021 %>%
    MEnvelope(r = 0:(10 * Distance)/5, ReferenceType = "Acer",
        NeighborType = "Acer", NumberOfSimulations = NumberOfSimulations) ->
    M_Acer
autoplot(M_Acer)
```



To map it, individual values must be calculated at the chosen distance, that is 15 meters.

```
trees_2021 %>%
    Mhat(r = c(0, Distance), ReferenceType = "Acer",
        NeighborType = "Acer", Individual = TRUE) ->
    M_ind_Acer
```

The map requires krigeing the individual values on a grid of points. To build the grid, the size ratio of the spatial window is calculated. The number of rows and columns of the grid will respect this ratio so that its points are equally spaced.

```
# Window ratio
ratio <- with(trees_infested$window, {
    (yrange[2] - yrange[1])/(xrange[2] - xrange[1])
})
# Map resolution: number of columns of the grid.
resolution <- 512
```

A kriged weighted, marked, planar point patterns (kwmppp) object is produced and plotted. Logged trees are added to the map:

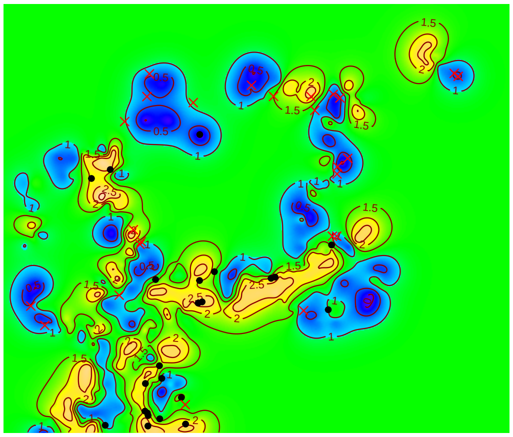
- infested trees are black points,

- Decaying trees are red crosses.

```
trees_2021 %>%
    kwmppp(fvind = M_ind_Acer, distance = Distance,
        ReferenceType = "Acer", Nbx = resolution, Nby = resolution *
            ratio) -> map_acer
```

```
## [using ordinary kriging]
```

```
plot(map_acer)
# Add infested trees
points(trees_infested[trees_infested$marks$PointType ==
    "infested"], pch = 20)
# And decaying trees
points(trees_infested[trees_infested$marks$PointType ==
    "Decaying"], pch = 4, col = "red")
```



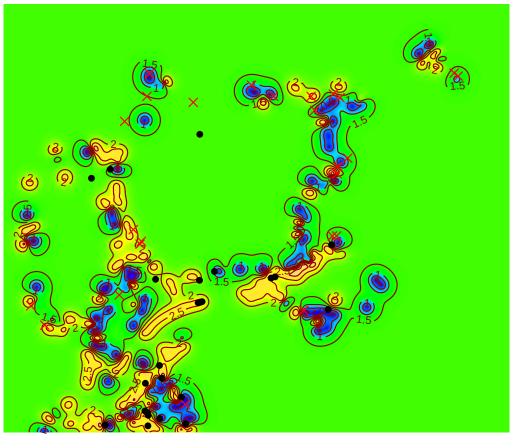
infested trees are present in the areas where maples are concentrated.

The 15-meter distance has been chosen because it is a plausible distance of contagion between trees. A shorter distance, say 6 meters, would be justified as a peak of M(distance) but neighbors are scarse at this scale, leading to less statistical power. At greater distances, such as 30m, concentration is still significant but neighborhoods overlap more, leading to less clear results.

Both alternative distances are tested here.

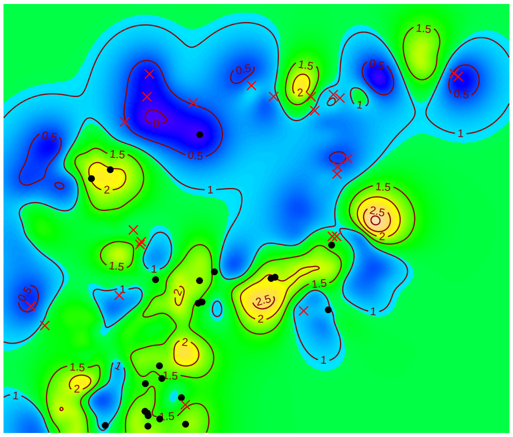
Considering 6-meter-radius neighborhoods, the map shows that some high and low values of local concentration are not detected:

```
## [using ordinary kriging]
```



30-meter-radius neighborhoods are shown below:

```
## [using ordinary kriging]
```

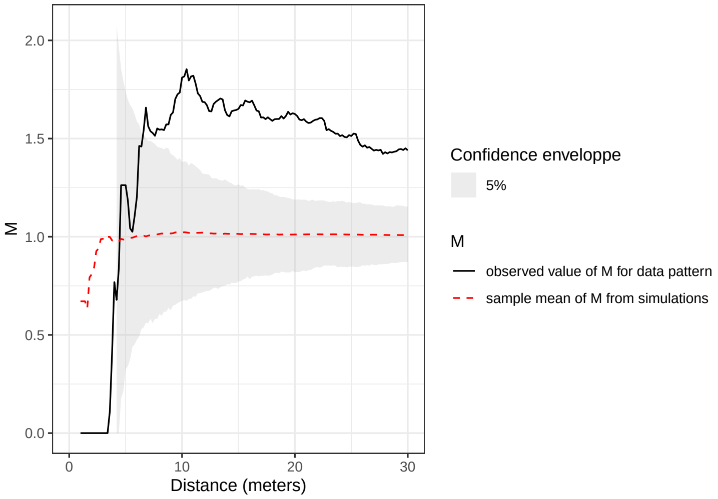


Increasing the distance that defines neighborhoods extends the areas where high or low concentration is detected around the most isolated trees, making them more visible, but blurs them in dense areas, where high and low values overlap and cancel out (e.g. in the south-eastern part of the map). So the appropriate distance should be chosen according to the knowledge of the studied process.

3.3 Concentration of maples around infested trees

To test the intertype concentration between sane and infested maple trees, the intertype M statistic is computed.

```
trees_infested %>%
    MEnvelope(r = 0:(10 * Distance)/5, ReferenceType = "infested",
        NeighborType = "Acer", NumberOfSimulations = NumberOfSimulations) ->
    M_infested_Acer
autoplot(M_infested_Acer)
```

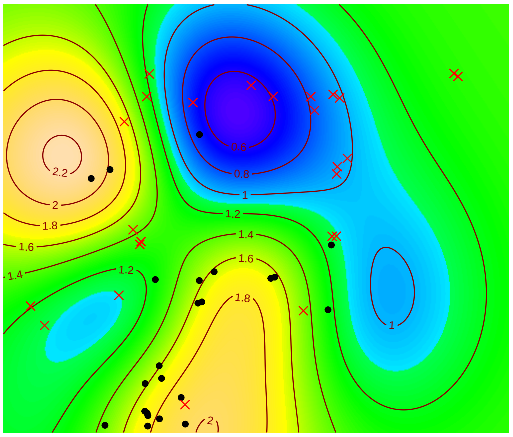


The map is produced.

```
trees_infested %>%
    Mhat(ReferenceType = "infested", NeighborType = "Acer",
        Individual = TRUE) -> M_ind_infested_Acer
trees_infested %>%
    kwmppp(fvind = M_ind_infested_Acer, distance = Distance,
        ReferenceType = "infested", Nbx = resolution,
        Nby = resolution * ratio) -> map_infested_acer
```

```
## [using ordinary kriging]
```

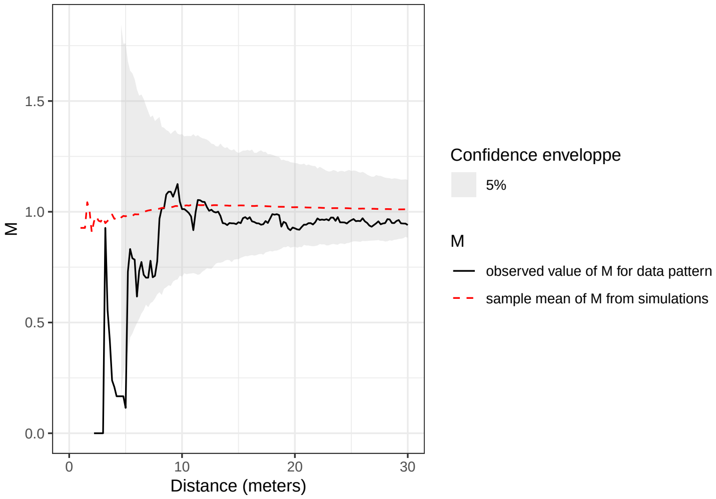
```
plot(map_infested_acer)
# Add infested trees
points(trees_infested[trees_infested$marks$PointType ==
    "infested"], pch = 20)
# And decaying trees
points(trees_infested[trees_infested$marks$PointType ==
    "Decaying"], pch = 4, col = "red")
```



infested and sane maple trees are significantly concentrated. The map shows that the local concentration of maples around infested trees is generally high (warm colors around black points).

In contrast, decaying trees and maples rather, but non significantly, repulse each others: the intertype M function between them is plotted below:

```
trees_infested %>%
    MEnvelope(r = 0:(10 * Distance)/5, ReferenceType = "Decaying",
        NeighborType = "Acer", NumberOfSimulations = NumberOfSimulations) ->
    M_Decaying_Acer
autoplot(M_Decaying_Acer)
```



References

Marcon, E., S. Traissac, F. Puech, and G. Lang (2015). Tools to characterize point patterns: Dbmss for R. *Journal of Statistical Software 67*(3), 1–15.

Matérn, B. (1960). Spatial variation. *Meddelanden från Statens Skogsforskningsinstitut 49*(5), 1–144.

R Core Team (2022). *R: A Language and Environment for Statistical Computing.* Vienna, Austria: R Foundation for Statistical Computing.