

Appendix

Florence Puech

Eric Marcon

December 10, 2022

Abstract

Code to reproduce the map of the main text.

1 Data

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

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
```

1.1.1 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.

1.1.2 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(especearbrepcedent), !is.na(libellefrancaisarbrepcedent),
    !is.na(genrearbrepcedent)) %>%
  filter(libellefrancaisarbrepcedent != "Non spécifié") %>%
  filter(especearbrepcedent != "n. sp.") %>%
  # Create a field with the species name
  mutate(Species_name = as.factor(paste(genrearbrepcedent,
    especearbrepcedent))) %>%
  # 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(genrearbrepcedent)) %>%
  mutate(Species = as.factor(especearbrepcedent)) %>%
  # Reason for logging (in French)
  mutate(Logging_reason = motifabattagearbrepcedent) %>%
  # Rename and finally select columns
  rename(French_species_name = libellefrancaisarbrepcedent) %>%
  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.

1.2 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
```

1.2.1 Simpler logging reasons

Logging reasons can be:

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

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

1.2.2 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)
```

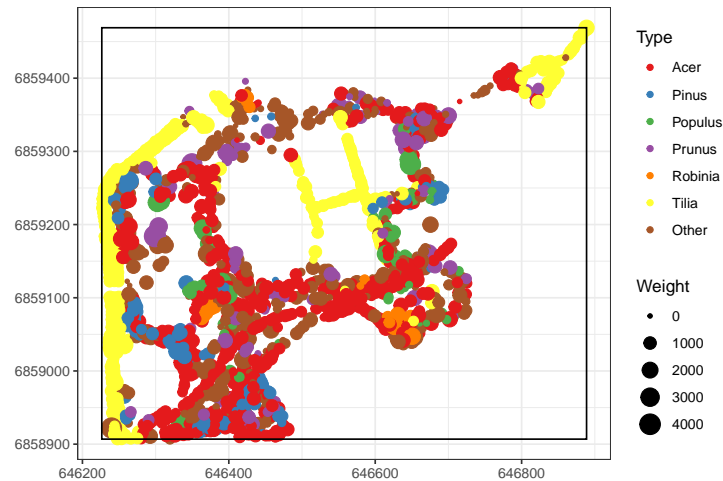
1.3 Point patterns

`dbmms` uses weighted, marked, planar point patterns (`wmppp`). A `wmppp` named `trees_infected` 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)) %>%
  wmpvp(window = as.owin(st_bbox(trees)), unitname = c("meter",
    "meters")) -> trees_infected
```

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)) %>%
  wmpvp(window = as.owin(st_bbox(trees_all)), unitname = c("meter",
    "meters")) -> trees_2021
autoplot(trees_2021)
```



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

2 Spatial analyses

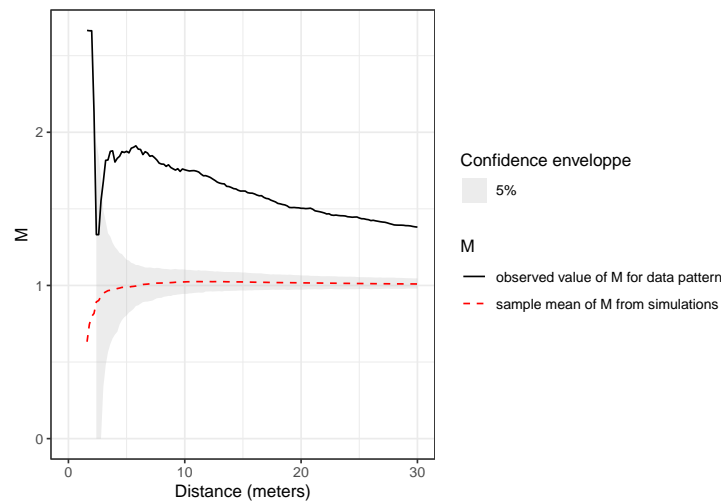
2.1 Spatial concentration of maple trees

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
```

```
## Generating 1000 simulations by evaluating
## expression ...
## 1, 2, 3, .....10.....20.....30.....40..
## .....50.....60.....70.....80....
## ....90.....100.....110.....120.....
## ...130.....140.....150.....160.....
## .170.....180.....190.....200.....210
## .....220.....230.....240.....250..
## .....260.....270.....280.....290...
## ...300.....310.....320.....330.....
## .340.....350.....360.....370.....
## .380.....390.....400.....410.....420
## .....430.....440.....450.....460..
## ....470.....480.....490.....500....
## ...510.....520.....530.....540.....
## ..550.....560.....570.....580.....
## .590.....600.....610.....620.....630
## .....640.....650.....660.....670..
## ...680.....690.....700.....710....
## ..720.....730.....740.....750.....
## .760.....770.....780.....790.....
## .800.....810.....820.....830.....840
## .....850.....860.....870.....880..
## ...890.....900.....910.....920....
## ..930.....940.....950.....960.....
## .970.....980.....990.....1000.
##
## Done.
```

```
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 kriging 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_infected>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:

- Infected 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

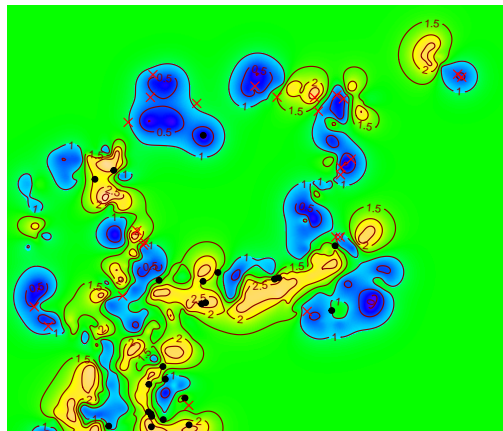
```

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

```

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

```



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

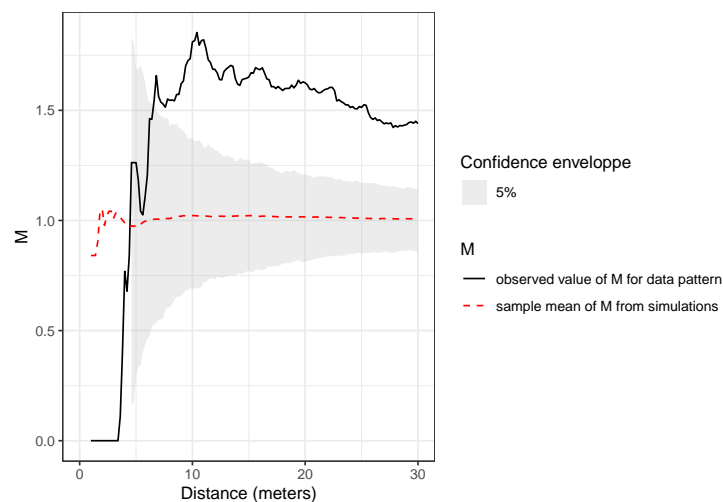
2.2 Concentration of maples around infected trees

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

```
trees_infected %>%
  MEnvelope(r = 0:(10 * Distance)/5, ReferenceType = "Infected",
    NeighborType = "Acer", NumberOfSimulations = NumberOfSimulations) ->
  M_Infected_Acer
```

```
## Generating 1000 simulations by evaluating
## expression ...
## 1, 2, 3, .....10.....20.....30.....40..
## .....50.....60.....70.....80....
## .....90.....100.....110.....120.....
## .....130.....140.....150.....160.....
## .....170.....180.....190.....200.....210
## .....220.....230.....240.....250...
## .....260.....270.....280.....290....
## .....300.....310.....320.....330.....
## .....340.....350.....360.....370.....
## .....380.....390.....400.....410.....420
## .....430.....440.....450.....460...
## .....470.....480.....490.....500....
## .....510.....520.....530.....540.....
## .....550.....560.....570.....580.....
## .....590.....600.....610.....620.....630
## .....640.....650.....660.....670...
## .....680.....690.....700.....710....
## .....720.....730.....740.....750.....
## .....760.....770.....780.....790.....
## .....800.....810.....820.....830.....840
## .....850.....860.....870.....880...
## .....890.....900.....910.....920....
## .....930.....940.....950.....960.....
## .....970.....980.....990.....1000..
##
## Done.
```

```
autoplot(M_Infected_Acer)
```

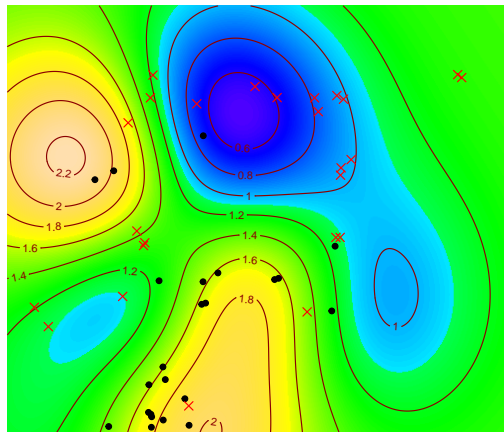


The map is produced.

```
trees_infected %>%
  Mhat(ReferenceType = "Infected", NeighborType = "Acer",
        Individual = TRUE) -> M_ind_Infected_Acer
trees_infected %>%
  kwmppp(fvind = M_ind_Infected_Acer, distance = Distance,
          ReferenceType = "Infected", Nbx = resolution,
          Nby = resolution * ratio) -> map_infected_acer
```

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

```
plot(map_infected_acer)
# Add infected trees
points(trees_infected[trees_infected$marks$PointType ==
  "Infected"], pch = 20)
# And decaying trees
points(trees_infected[trees_infected$marks$PointType ==
  "Decaying"], pch = 4, col = "red")
```



Infected and sane maple trees are significantly concentrated.
In contrast, decaying trees and mapples rather repulse each others.

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

```
## Generating 1000 simulations by evaluating
## expression ...
## 1, 2, 3, .....10.....20.....30.....40..
## .....50.....60.....70.....80....
## ...90.....100.....110.....120.....
## ...130.....140.....150.....160.....
## ...170.....180.....190.....200.....210
## .....220.....230.....240.....250..
## .....260.....270.....280.....290....
## .....300.....310.....320.....330.....
## ...340.....350.....360.....370.....
```



```
## .380.....390.....400.....410.....420
## .....430.....440.....450.....460..
## .....470.....480.....490.....500...
## .....510.....520.....530.....540....
## ...550.....560.....570.....580.....
## .590.....600.....610.....620.....630
## .....640.....650.....660.....670..
## .....680.....690.....700.....710...
## ...720.....730.....740.....750.....
## ...760.....770.....780.....790.....
## .800.....810.....820.....830.....840
## .....850.....860.....870.....880..
## .....890.....900.....910.....920....
## ...930.....940.....950.....960.....
## ..970.....980.....990.....1000.
##
## Done.
```

```
autoplot(M_Decaying_Acer)
```

