NFI process: grid-based sampling

2024-07-05

```
library(readxl) # to read the initial file
library(magrittr) # for pipes
library(tibble) # for add_column
```

Preliminaries

```
# Importing the database with the artificial forest
trees <- read_excel("~/work/Sim_article2024/Data/artificial_forest_round.xls")
```

```
# Functions to produce a local variable
source("~/work/Sim_article2024/Useful_functions/Circle.R")
```

A first sample to produce a first estimation

Sampling in one 10th of the small squares of a grid

```
### Randomizing the positions of the numbers (from 1 to 10)
# used to identify small squares in the grid
random10 <- sample(1:10)
i <- sample(1:10, 1) # choice of the rank of the identifier we want
nb <- random10[i] # the identifier thus chosen
```

```
# Dimension of a small square (length of one side of the square)
# We assume that c_dim is a divisor of 1000
c_dim <- 10
```

```
# Now, we sample at random following the grid
random_points <- data.frame(x = numeric(), y = numeric())

for (l in 0:((1000/c_dim)-1)) {
  if (l==0){
    x_init <- i*c_dim
  }else if ((3*l+i)*c_dim < (10)*c_dim){
    x_init <- (3*l+i)*c_dim
  }else{
    x_init <- (((3*l+i)*c_dim)%%(c_dim*10))
  }
  k <- 0
  while (x_init + 10*c_dim*k < 1000) {
    a <- runif(1, x_init + 10*c_dim*k, x_init + 10*c_dim*k + c_dim)
    b <- runif(1, l*c_dim, (l+1)*c_dim)
    random_points <- rbind(random_points, list(a,b))
    k <- k+1
  }
```

```
}
colnames(random_points) <- list("x", "y")
```

```
plot(x = random_points$x, y = random_points$y, xlab = "", ylab = "", main = "Random points", panel.firs
```

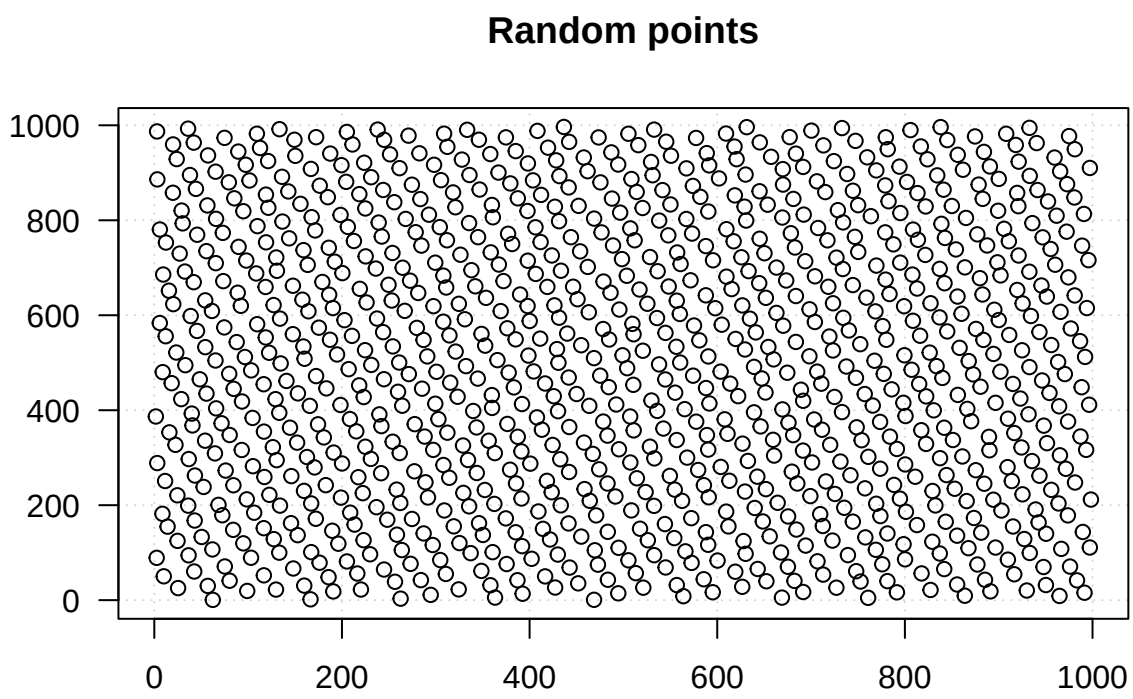

Figure 1. **Points sampled following a grid over the whole territory**.

```
rm(a,b,l,x_init,k)
```

```
### Values available after sampling

# Sample size
n <- nrow(random_points)

# Marginal density: density of the uniform distribution over 1:1000
# Which leads us to the inclusion density function
p <- sum(rep((1/1000)*(1/1000), n))
```

Estimations based on the sample previously made

```
# Number of trees
a <- mapply(circle3_nb, random_points[, 1], random_points[, 2], MoreArgs = list(15, 9, 6))
est_init_nb <- sum(a/p)
rm(a)
cat("Number of trees:", nrow(trees), "\nEstimation of the number of trees:", est_init_nb)
```

```
## Number of trees: 30942
```

```
## Estimation of the number of trees: 29471.57
```

```
# Total volume of trees
a <- mapply(circle3_vol, random_points[, 1], random_points[, 2], MoreArgs = list(15, 9, 6))
est_init_vol <- sum(a/p)
rm(a)
cat("Volume of the forest:", sum(trees$v), "\nEstimation of the volume of the forest:", est_init_vol)
```

```
## Volume of the forest: 5412.722
## Estimation of the volume of the forest: 5451.174
```

Bootstrap to produce confidence intervals

Creating other samples based on the previous one

```
# Number of bootstrap samples
B <- 500
```

```
f_boot <- function(df_sample, nboot){
  # df_sample: original sample
  # nboot: number of bootstrap samples made thanks to df_sample

  df_boot <- tibble(.rows = nrow(df_sample))
  for (b in 1:nboot) {
    nb <- sample(1:nrow(df_sample), nrow(df_sample), replace = TRUE)
    sample_b <- df_sample[nb,]
    df_boot <- df_boot %>% add_column(new_col = sample_b)
    colnames(df_boot)[b] <- paste("ech", b, sep = "")
  }
  return(do.call(data.frame, df_boot))
  # Return a data frame with nboot samples of 2 variables and of size n
}
```

```
# Making of bootstrap samples
df_boot <- f_boot(random_points, B)
```

Estimations and confidence intervals based on the bootstrap samples with three circles

Number of trees

```
# Making of one estimation for each sample made by the bootstrap method
est_boot <- function(df_boot, fun){
  # df_boot: dataframe with 2*B columns and n rows
  # fun: function used to calculate the local variable
  # based on one point of the sample

  nboot <- length(df_boot)/2
  vec_est <- c()
  for (c in 1:nboot) {
    a <- mapply(fun, df_boot[, (2*c)-1], df_boot[, 2*c], MoreArgs = list(15, 9, 6))
    vec_est <- append(vec_est, sum(a/p))
  }
  vec_est <- sort(vec_est)
  return(vec_est)
```

```
  # Vector with one estimation for each sample
}
```

```
# Making of one estimation for each sample made by the bootstrap method
est_boot_nb <- est_boot(df_boot, circle3_nb)
```

```
ci_perc <- function(est_init, est_boot){
  # est_init: estimation based on the initial sample
  # est_boot: estimations based on bootstrap samples

  B <- length(est_boot)

  ## 1st step: percentile CI
  lim_inf <- est_boot[floor(B*2.5/100)]
  lim_sup <- est_boot[B - floor(B*2.5/100)]
  cat("95% confidence interval using the percentile method:
      [", lim_inf, ";", lim_sup, "]\n" )

  ## 2nd step: reverse percentile CI
  cat("95% confidence interval using the reverse percentile method:
      [", est_init*2 - lim_sup, ";", est_init*2 - lim_inf, "]\n" )
  return(c(est_init,lim_inf,lim_sup))
}
```

```
print("Two intervals for the number of trees:")
```

```
## [1] "Two intervals for the number of trees:"
```

```
res_nb <- ci_perc(est_init_nb,est_boot_nb)
```

```
## 95% confidence interval using the percentile method:
##       [ 26495.96 ; 32368.78 ]
## 95% confidence interval using the reverse percentile method:
##       [ 26574.36 ; 32447.17 ]
```

```
# Visualizing the density, the interval and the first estimation
plot(density(est_boot_nb), xlab="Number of trees",ylab="", main = "Distribution based on bootstrap", las
abline(v=res_nb[2], col="blue")
abline(v=res_nb[3], col="blue")
abline(v=est_init_nb*2 - res_nb[3], col="green")
abline(v=est_init_nb*2 - res_nb[2], col="green")
abline(v=est_init_nb, col="red")
abline(v=nrow(trees), col="black")
```

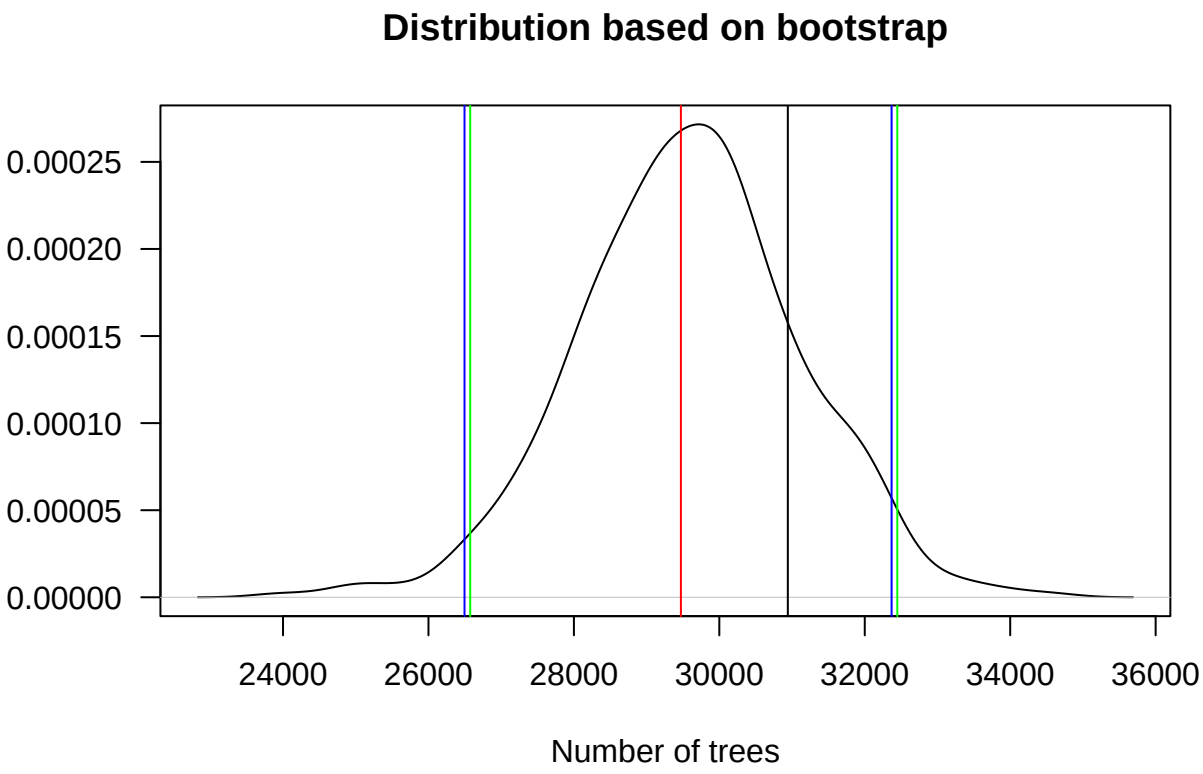

Figure 2. **Density based on bootstrap**. The true value (in black), the initial estimation (in red), percentile interval (in blue), reverse percentile interval (in green).

Total volume of the forest

```
# Making of one estimation for each sample made by the bootstrap method
est_boot_vol <- est_boot(df_boot, circle3_vol)
```

```
print("Two intervals for the volume of the forest:")
```

```
## [1] "Two intervals for the volume of the forest:"
```

```
res_vol <- ci_perc(est_init_vol,est_boot_vol)
```

```
## 95% confidence interval using the percentile method:
##       [ 4887.976 ; 6022.414 ]
## 95% confidence interval using the reverse percentile method:
##       [ 4879.935 ; 6014.373 ]
```

```
# Visualizing the density, the interval and the first estimation
plot(density(est_boot_vol), xlab="Volume of the forest",ylab="", main = "Distribution based on bootstrap
abline(v=res_vol[2], col="blue")
abline(v=res_vol[3], col="blue")
abline(v=est_init_vol*2 - res_vol[3], col="green")
abline(v=est_init_vol*2 - res_vol[2], col="green")
abline(v=est_init_vol, col="red")
abline(v=sum(trees$v), col="black")
```

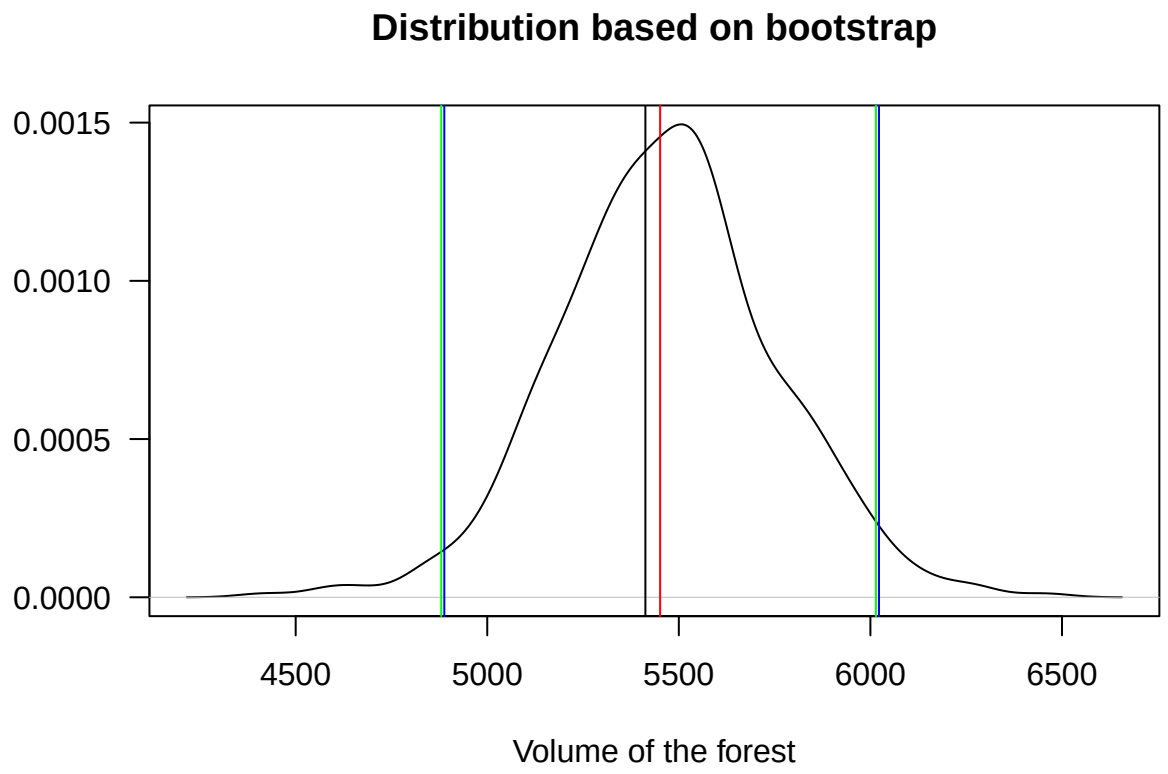

Figure 3. **Density based on bootstrap**. The true value (in black), the initial estimation (in red), percentile interval (in blue), reverse percentile interval (in green).