Assignment 3 - Stochastic processes and Bayesian inference (MVE550)

Luca Modica
Hugo Manuel Alves Henriques e Silva
Linus Haraldsson

December 15, 2023

Part 1

Question (a)

(1) $f(x, y, \theta_1, \theta_2, \theta_3) = \dfrac{\exp\left(e^{\theta_1} x + e^{\theta_2} (y - \theta_3)^2\right) - 1}{\exp\left(e^{\theta_1} x + e^{\theta_2} (y - \theta_3)^2\right) + 1}$

a)

Likelihood:

$$\pi(\text{data} \mid \theta) = \pi(\text{data} \mid \theta_1, \theta_2, \theta_3)$$

$$= \prod_{i=1}^{N} f(x_i, y_i, \theta_1, \theta_2, \theta_3)^{z_i} \times \left[1 - f(x_i, y_i, \theta_1, \theta_2, \theta_3)\right]^{1 - z_i}$$

As we consider a flat prior for the parameters $\theta_1, \theta_2, \theta_3$, the posterior density is proportional to the likelihood. (uniform prior)

$$\text{Posterior}(\theta_1, \theta_2, \theta_3 \mid \text{data}) \propto \pi(\text{data} \mid \theta_1, \theta_2, \theta_3)$$

The likelihood corresponds to the product of the probability of every single observation as they are assumed to be independent from each other.

As z_i is a binary variable which indicates the presence or absence of a disease we have the case for the probability of a single observation

→ Presence $(z_i = 1)$

$\pi(\text{obs} \mid \theta_1, \theta_2, \theta_3) = f(x_i, y_i, \theta_1, \theta_2, \theta_3)$ which is the exact probability of developing the disease given the parameters.

→ Absence $(z_i = 0)$

$\pi(\text{obs} \mid \theta_1, \theta_2, \theta_3) = 1 - f(x_i, y_i, \theta_1, \theta_2, \theta_3)$ which is the probability of not developing the disease given the parameters

Question (b)

The required R function is displayed below:

```r
log_posterior <- function(theta1, theta2, theta3, data) {
  x <- data[,1]
  y <- data[,2]
  z <- data[,3]

  # Compute the likelihood for each observation

  # due to the use of the logarithm get that
  # log(f^z[i])) = z[i] * log(f)
  # log((1 - f)^(1 - z[i])) = (1 - z[i]) * log(1 - f)

  epsilon <- 1e-8 # Small constant to prevent log(0)
  likelihoods <- sapply(1:length(x), function(i) {
    f <- (exp(exp(theta1) * x[i] + exp(theta2) * (y[i] - theta3)^2) - 1) /
      (exp(exp(theta1) * x[i] + exp(theta2) * (y[i] - theta3)^2) + 1)
    z[i] * log(f + epsilon) + (1 - z[i]) * log(1 - f + epsilon)
  })

  # the product of the likelihoods of each observation log(a*b*...*N)
  # is equivalent to the sum of the log of each observation's likelihood
  # log(a) + log(b) + ... + log(N)
  return(sum(likelihoods))
}
```

Question (c)

For starting values for the chain we studied the values for θ that might be reasonable for the given data.

Pollutant Concentration (x)

- Mean: ≈ 4.59
- Standard Deviation: ≈ 2.71
- Range: 0.81 to 12.26

Temperature (y)

- Mean: ≈ 17.22
- Standard Deviation: ≈ 4.62
- Range: 10.06 to 24.92

Disease Presence (z)

- Mean: ≈ 0.58
- This is a binary variable, indicating the presence (1) or absence (0) of the disease.

For θ_1 we analyse pollutant concentration. The range of x is wide, therefore we want to start with a small value to see how sensitive the model could be to changes of pollutant concentration. We will start with a value close to zero.

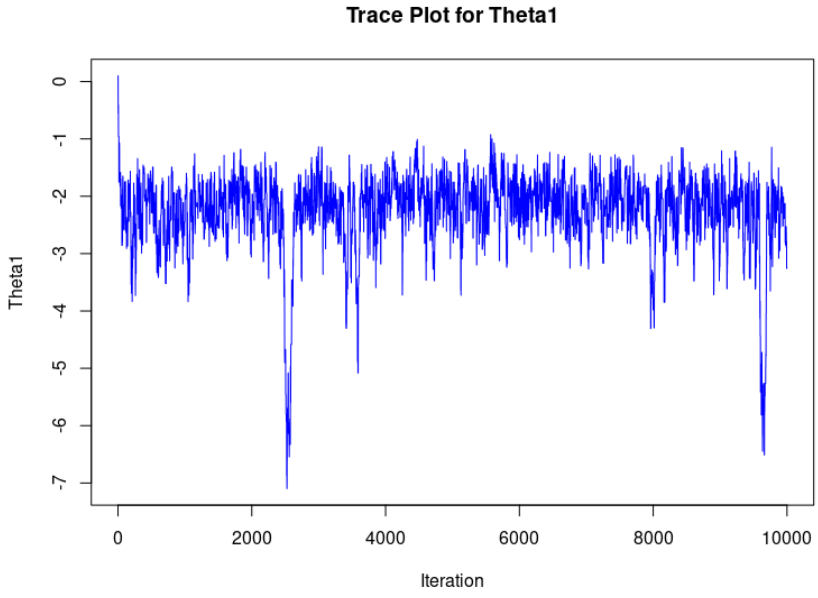


Figure 1: Trace plot mapping simulated values for θ_1.

For θ_2 the same will be done. The temperature also has a wide range, so a value close to zero will be used.

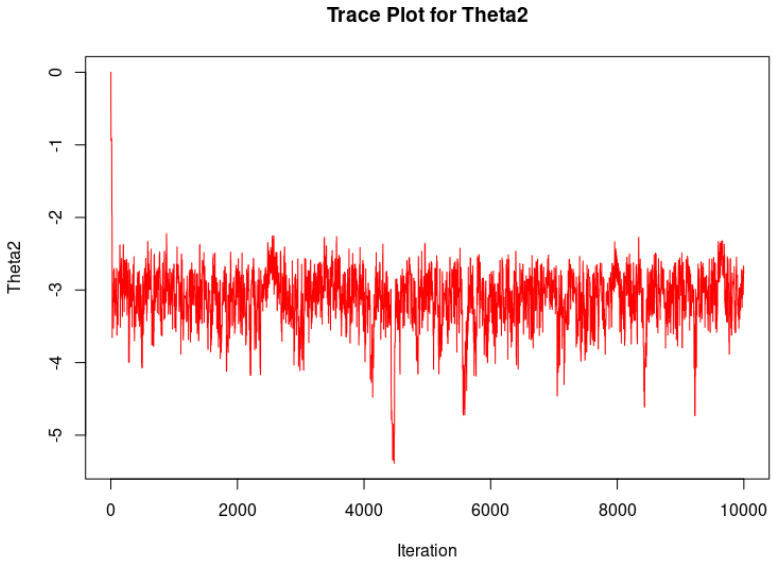


Figure 2: Trace plot mapping simulated values for θ_2.

Finally, θ_3 is associated with temperature shift, so given that the mean of the temperature is around the value of 17.22, a starting point somewhere around this value would make sense. It would work as a baseline around which temperature variations affect disease probability.

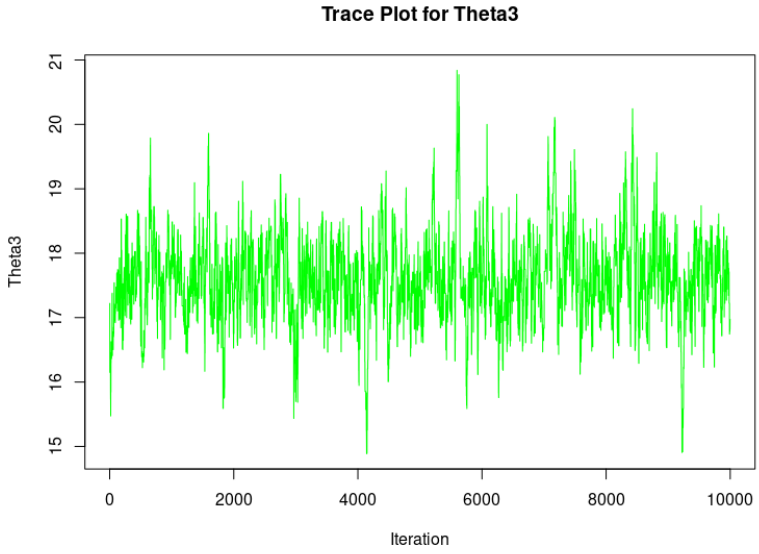


Figure 3: Trace plot mapping simulated values for θ_3.

Code used to implement the MCMC algorithm and realizing the simulations:

```r
#c)
data <- read.table("dataAssignment3.txt", header = TRUE)

# Calculate mean, standard deviation, and range for each column
mean_x <- mean(data$x)
std_dev_x <- sd(data$x)
range_x <- range(data$x)

mean_y <- mean(data$y)
std_dev_y <- sd(data$y)
range_y <- range(data$y)

mean_z <- mean(data$z)
# For binary variable z, standard deviation and range may be less informative
range_z <- range(data$z)

# Print the calculated values
print(paste("Mean of x:", mean_x, "Standard Deviation of x:", std_dev_x, "Range of x:",
    paste(range_x, collapse = " to ")))
print(paste("Mean of y:", mean_y, "Standard Deviation of y:", std_dev_y, "Range of y:",
    paste(range_y, collapse = " to ")))
print(paste("Mean of z:", mean_z, "Range of z:", paste(range_z, collapse = " to ")))


mcmc <- function(start_values, data, iterations = 10000) {
  current_theta <- start_values
  chain <- matrix(NA, nrow = iterations, ncol = 3)

  for (i in 1:iterations) {
    proposed_theta <- current_theta + rnorm(3, mean = 0, sd = 0.4)

    current_log_posterior <- log_posterior(current_theta[1], current_theta[2],
        current_theta[3], data)
    proposed_log_posterior <- log_posterior(proposed_theta[1], proposed_theta[2],
        proposed_theta[3], data)

    if (!is.finite(current_log_posterior) || !is.finite(proposed_log_posterior)) {
      next  # Skip iteration if log-posterior is not finite
    }

    #symmetric random walk
    acceptance_ratio <- exp(proposed_log_posterior - current_log_posterior)

    if (runif(1) < acceptance_ratio) {
      current_theta <- proposed_theta
    }

    chain[i, ] <- current_theta
  }

  return(chain)
}
```

```r
# Example usage
start_values <- c(0.1, 0, 17.22) # Replace with your starting values
chain <- mcmc(start_values, data)


# Assuming 'chain' is the output from your MCMC function

# Plot for theta1
plot(chain[, 1], type = "l", col = "blue", xlab = "Iteration", ylab = "Theta1", main =
    "Trace Plot for Theta1")

# Plot for theta2
plot(chain[, 2], type = "l", col = "red", xlab = "Iteration", ylab = "Theta2", main = "Trace
    Plot for Theta2")

# Plot for theta3
plot(chain[, 3], type = "l", col = "green", xlab = "Iteration", ylab = "Theta3", main =
    "Trace Plot for Theta3")
```

```
[1] Mean of x: 4.5909819069051 Standard Deviation of x: 2.71111606954208 Range of x:
    0.812655679677847 to 12.2625424363298
[1] Mean of y: 17.2235614431819 Standard Deviation of y: 4.62067865839263 Range of y:
    10.0599946908187 to 24.9233423965052
[1] Mean of z: 0.580645161290323 Range of z: 0 to 1
```

Question (d)

With the R code displayed below, we computed numerically the required predicted probabilities:

```r
# d)
# Assuming 'chain' contains your MCMC samples
theta_mean <- colMeans(chain)

# Extracting mean values of theta1, theta2, theta3
theta1_mean <- theta_mean[1]
theta2_mean <- theta_mean[2]
theta3_mean <- theta_mean[3]

# Probability function
f <- function(x, y, theta1, theta2, theta3) {
  (exp(exp(theta1) * x + exp(theta2) * (y - theta3)^2) - 1) /
    (exp(exp(theta1) * x + exp(theta2) * (y - theta3)^2) + 1)
}

# Compute the probability for one animal
p_single_animal <- f(3, 13, theta1_mean, theta2_mean, theta3_mean)

# Compute the probability that 9 out of 10 animals will develop the disease
# binomial(9; 10, p)
p_nine_out_of_ten <- choose(10, 9) * p_single_animal^9 * (1 - p_single_animal)^1

# Print the results
print(paste("Probability for one animal:", p_single_animal))
print(paste("Probability for 9 out of 10 animals:", p_nine_out_of_ten))
```

```
[1] Probability for one animal: 0.558651299021837
[1] Probability for 9 out of 10 animals: 0.0233910204866009
```

Part 2

Question (a)

Since we are considering a spatial Poisson Process, the related parameter is computed as follows:

$$\lambda|\text{area\_square}| = 36 \cdot |[0.2, 0.6] \times [0.2, 0.6]| = 5.72.$$

The probability that there are 6 trees or more in that area is then computed as follows:

$$P(N \geq 6) + 1 - P(N \leq 5) = 1 - e^{-5.76} \cdot \sum_{i=0}^{5} \frac{\lambda^i}{i!} = 0.515.$$

Question (b)

Since we are considering 2 areas $A = [0.2, 0.6] \times [0.2, 0.6]$ and $B = [0.4, 0.8] \times [0.4, 0.8]$ that overlap to each other, we also need to take into account the overlapping area, which is $C = [0.4, 0.6] \times [0.4, 0.6]$. In this way we can consider the disjoint sets C, $A - C$ and $B - C$ and the related independent random variables N_C, N_{A-C} and N_{B-C}, with the respective lambda parameter:

$$\begin{aligned}
\lambda_C &= \lambda|C| = 36 \cdot 0.4 = 1.44, \\
\lambda_{A-C} &= \lambda_A - \lambda_C = 5.76 - 1.44 = 4.32, \\
\lambda_{B-C} &= \lambda_B - \lambda_C = 5.76 - 1.44 = 4.32.
\end{aligned}$$

We can compute the probability that there are exactly 4 trees in the square A and at the same time exactly 4 trees in the square B as follows. The solution will take into account all the possible scenario, considering trees in the common area or in one 2 specified squares.

$$\begin{aligned}
&P(N_A = 4, N_B = 4) \\
&= \sum_{i=0}^{4} P(N_C = i) P(N_{A-C} = 4 - i) P(N_{B-C} = 4 - i) \\
&= \sum_{i=0}^{4} (e^{-\lambda_C} \frac{\lambda_C^i}{i!})(e^{-\lambda_{A-C}} \frac{\lambda_{A-C}^{4-i}}{4-i!})(e^{-\lambda_{B-C}} \frac{\lambda_{A-C}^{4-i}}{4-i!}) \\
&= 0.02390222.
\end{aligned}$$

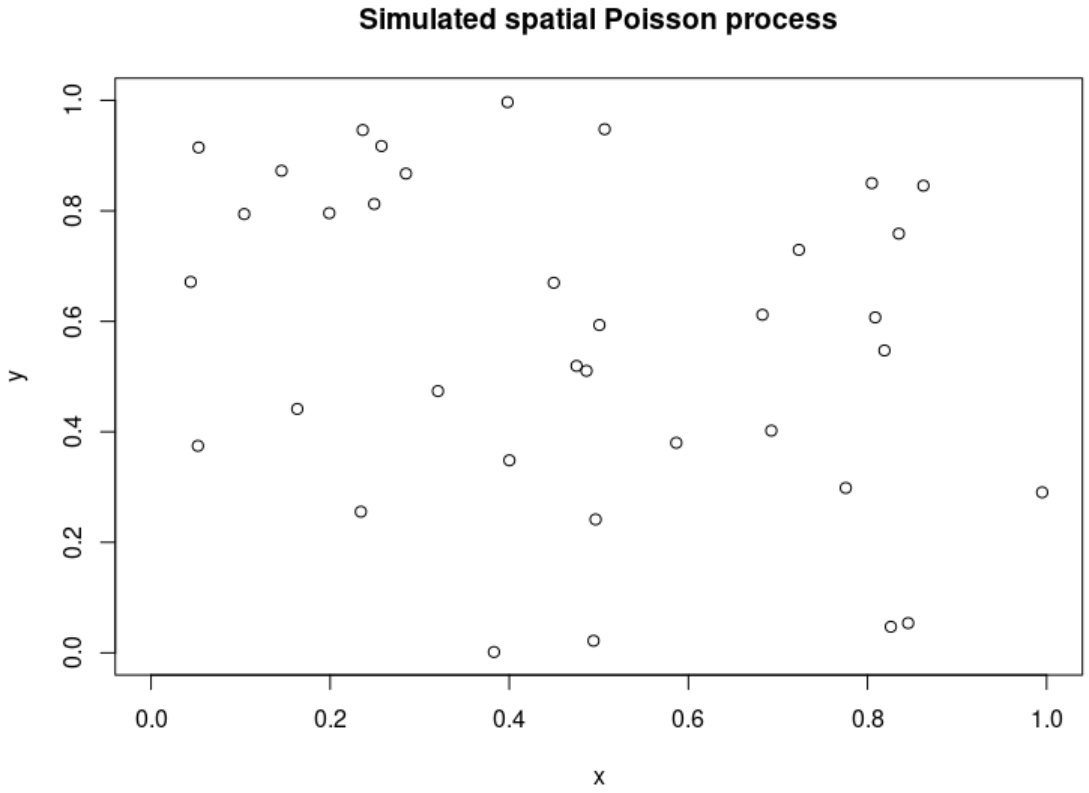
Question (c)

Code to simulate the mentioned spatial Poisson process:

```
# question 2c: simulation
spatial_poisson_sim <- function(lambda, trials) {
    simlist <- numeric(trials)
    for (i in 1:trials){
      N <- rpois(1, lambda)
      x <- runif(N,0,1)
      y <- runif(N,0,1)
      simlist[i] = N
    }
```

```
  N_mean = round(mean(simlist))
  print("mean number of trees after the simulations: ")
  print(N_mean)
  x <- runif(N_mean,0,1)
  y <- runif(N_mean,0,1)
  plot(x, y, xlim=c(0,1), ylim=c(0,1), main="Simulated spatial Poisson process")
}
spatial_poisson_sim(36, 10000)
```

Example figure of one simulation:



Question (c)

Based on the data we have about the observation and the proportion of the prior, we can derive the posterior of λ by looking to the *Poisson - Gamma conjugacy*. In other words:

$$\lambda|\text{data} \sim \text{Gamma}(0+36, 0+1),$$
$$\pi(\lambda|\text{data}) \propto_\lambda \text{Gamma}(\lambda, 36, 1).$$

The code that create a simulation which uses this posterior instead of a fixed λ is the following:

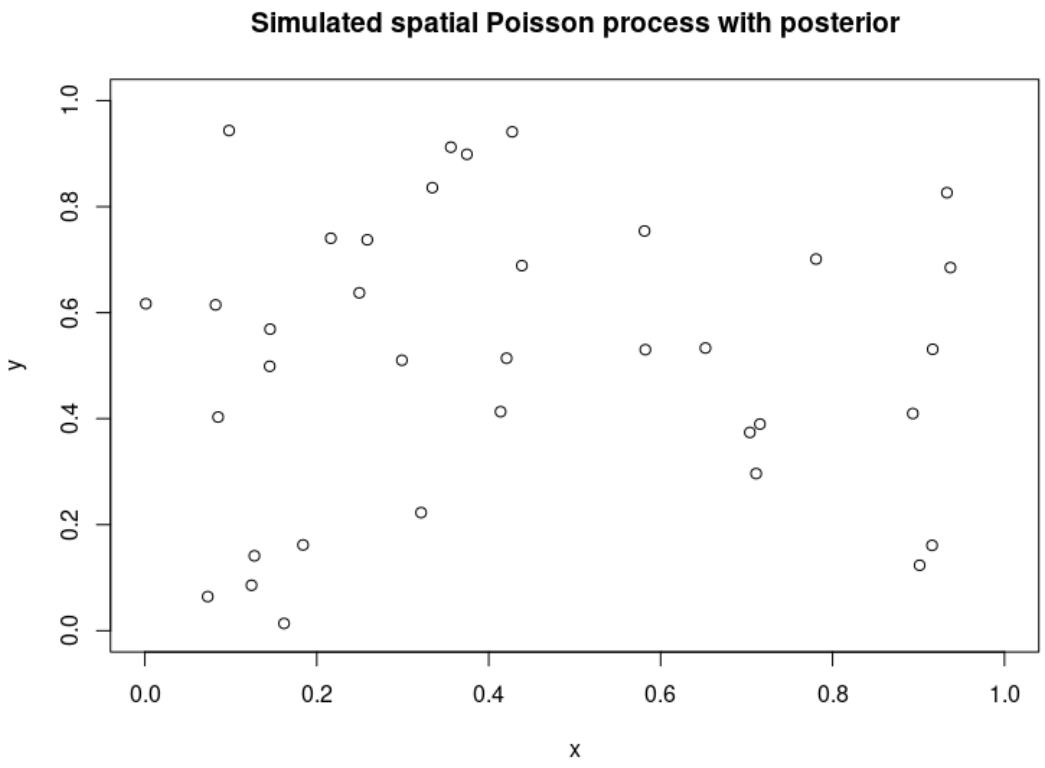
```
# question 2d: simulation with posterior
```

```r
# Define the prior parameters for the Gamma distribution
spatial_poisson_sim_posterior <- function(trials) {
  # Update the posterior parameters based on the observed data
  alpha_post <- 0 + 36 # 36 trees observed
  beta_post <- 0 + 1
  simlist <- numeric(trials)

  for (i in 1:trials) {
    N <- rgamma(1, shape = alpha_post, rate = beta_post)
    x <- runif(N,0,1)
    y <- runif(N,0,1)
    simlist[i] = N
  }

  N_mean = round(mean(simlist))
  print("mean number of trees after the simulations (with posterior): ")
  print(N_mean)
  x <- runif(N_mean,0,1)
  y <- runif(N_mean,0,1)
  plot(x, y, xlim=c(0,1), ylim=c(0,1), main="Simulated spatial Poisson process with
      posterior")
}
spatial_poisson_sim_posterior(10000)
```

Example figure of one simulation:



Question (e)

The R code below is to simulate the random variable Z, which represents the average over all points of the distance from this point to its nearest neighbour.

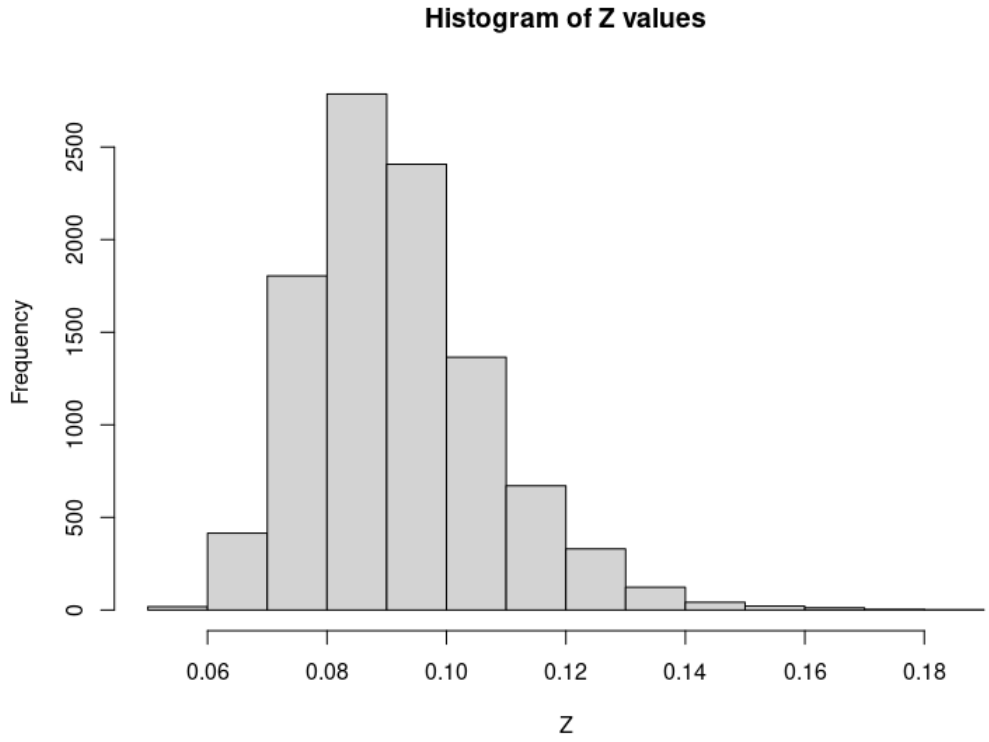
```r
# question 2e: simulation from Z
compute_Z <- function() {
  alpha_post <- 0 + 36
  beta_post <- 0 + 1
  l <- rgamma(1, shape = alpha_post, rate = beta_post)
  n <- rpois(1, l)
  x <- runif(n, 0, 1)
  y <- runif(n, 0, 1)

  # Compute Z
  distances <- matrix(NA, n, n)
  for (i in 1:n) {
    for (j in 1:n) {
      distances[i, j] <- sqrt((x[i] - x[j])^2 + (y[i] - y[j])^2)
    }
  }

  # Exclude self-distances (which are 0) by setting them to Inf
  diag(distances) <- Inf
  return (mean(apply(distances, 1, min)))
}

simulation_Z <- function(trials) {
  Z_values <- replicate(trials, compute_Z())
  hist(Z_values, main="Histogram of Z values", xlab="Z")
}
simulation_Z(10000)
```

Histogram of a random sample from the distribution of Z:



Question (f)

Using the result mentioned in the question and the result obtained in the previous one, we can say that the Poisson model is not a good model for these tree data. This conclusion is drawn by verifying that a Z value of 0.1358 is an outlier, as we can also notice in the histogram of the previous question.

To let that specific value of Z fit into the model, what should be done is to opt to a more complex spatial process. This in order to take into account specific factors related to the trees which can influence their positions (such as soil quality, water availability, ...), and to model eventual repulsion, clustering and spatial co-varieties of the trees themselves.