Tests for an Increasing Trend in the Intensity of a Poisson Process: A Power Study

5 ModIA

2024/2025

Introduction

In this study, we investigate two statistical tests for detecting an increasing trend in the intensity of a Nonhomogeneous Poisson Process (NHPP):
1. Laplace Test
2. Boswell's Likelihood Ratio Test

We use Monte Carlo simulations to generate data under different scenarios and estimate the power of these tests.

Theoretical Background

Laplace Test

The core idea of the Laplace test is to analyze the distribution of the arrival times $T_1, T_2, \ldots, T_n$ within the observation interval $[0, T^*]$. The test is based on the Laplace statistic F defined as the average of the arrival times:

$$F = \frac{1}{T^*} \sum_{i=1}^{n} T_i.$$

Under the null hypothesis of a constant intensity, the T_i are uniformly distributed. If the intensity is increasing, the T_i cluster towards the end, making F larger. We reject the null hypothesis if F is larger than a critical value.

Boswell Test

The core idea of the Boswell test is to compare the likelihood of the observed arrival times $T_1, T_2, \ldots, T_n$ under the assumption of a constant intensity (null hypothesis) versus a non-decreasing intensity (alternative hypothesis). The test is based on the likelihood ratio statistic W, which measures the difference between the maximum likelihoods under these two hypotheses:

$$W = 2 \left(\sum_{i=1}^{n} \log(\hat{\lambda}(T_i)) + n \log \left(\frac{T^*}{n} \right) \right),$$

where $\hat{\lambda}(T_i)$ is the optimal non-decreasing estimate of the intensity. Under the null hypothesis, W follows a chi-squared distribution. We reject the null hypothesis if W is larger than a critical value, indicating a better fit for a non-decreasing intensity.

Numerical Simulations

Simulation Methodology

To simulate arrival times $T_1, T_2, \ldots, T_n$ for an NHPP with a given intensity function $\lambda(t)$ over $[0, T^*]$, we use the inverse transform method.

```r
# Define the simulation function for a Poisson Process
simulate_hpp <- function(n, T_star) {
  return(sort(runif(n, min = 0, max = T_star)))
}
```

```r
# Define the simulation function for a Nonhomogeneous Poisson Process
simulate_nhpp <- function(n, T_star, lambda_func, lambda_inv_func) {
  u <- sort(runif(n, min = 0, max = lambda_func(T_star)))
  u <- lambda_inv_func(u)
  return(u)
}
```

To compute the power of the Laplace and Boswell tests, we perform a Monte Carlo simulation for different scenarios of increasing intensity functions $\lambda(t)$.

```r
# Define the Monte Carlo simulation function
monte_carlo_simulation <- function(n, T_star, lambda_func, lambda_inv_func,
                                   test_func, threshold, n_sim = 5000) {
  count_reject <- 0
  for (i in 1:n_sim) {
    # Simulate NHPP data
    times <- simulate_nhpp(n, T_star, lambda_func, lambda_inv_func)

    # Perform the test
    test_stat <- test_func(times, T_star, n)

    # Check if the test statistic exceeds the critical value
    if (test_stat > threshold) {
      count_reject <- count_reject + 1
    }
  }
  return(count_reject / n_sim)
}
```

We simulate data with various increasing intensity functions $\lambda(t)$:

- Exponential trend: $\begin{cases} \lambda(t) = e^{\beta t} \\ \Lambda(t) = \frac{1}{\beta}(e^{\beta t} - 1) \\ \Lambda^{-1}(u) = \frac{1}{\beta}\log(\beta u + 1) \end{cases}$
- Weibull trend: $\begin{cases} \lambda(t) = \beta t^{\beta - 1} \\ \Lambda(t) = t^{\beta} \\ \Lambda^{-1}(u) = u^{1/\beta} \end{cases}$

```r
# Define the intensity functions, the cumulative intensity functions and their inverses
# Exponential trend
lambda_exp <- function(beta) {
  return(function(t) { exp(beta * t) })
}
lambda_cum_exp <- function(beta) {
  return(function(t) { (1 / beta) * (exp(beta * t) - 1) })
}
lambda_inv_exp <- function(beta) {
  return(function(u) { (1 / beta) * log(beta * u + 1) })
}

# Weibull trend
lambda_weibull <- function(beta) {
  return(function(t) { beta * t^(beta - 1) })
}
lambda_cum_weibull <- function(beta) {
  return(function(t) { t^beta })
}
lambda_inv_weibull <- function(beta) {
  return(function(u) { u^(1 / beta) })
}
```

We define the Laplace and Boswell tests for an increasing intensity trend.

```r
lambda_hat <- function(PPi, T_star, n) {
  times = c(PPi, T_star)
  lambda_vals = numeric(n)

  # Compute the lambda_hat values
  for (k in 1:n) {

    lambda_val_max <- -Inf

    for (a in 1:k) {

      lambda_val_min <- Inf

      for (b in k:n) {
        lambda_val_min <- min(lambda_val_min, (b - a + 1) / (times[b + 1] - times[a]))
      }

      lambda_val_max <- max(lambda_val_max, lambda_val_min)
    }

    lambda_vals[k] <- lambda_val_max
  }
  return(lambda_vals)
}
```

```r
# Define the Laplace test
laplace_test <- function(times, T_star, n) {
  # Compute the Laplace statistic
```

```
  F <- sum(times) / T_star

  # Standardize the Laplace statistic
  Z <- (F - (n / 2)) * sqrt(12 / n)

  return(Z)
}

# Define the Boswell test
boswell_test <- function(times, T_star, n) {
  # Compute the optimal non-decreasing estimate of the intensity
  lambda_vals <- lambda_hat(times, T_star, n)

  # Compute the likelihood ratio statistic
  W <- 2 * (sum(log(lambda_vals)) + n * log(T_star / n))

  return(W)
}
```

We set the critical values for the Laplace and Boswell tests based on the significance level α.

```
stirling_number <- function(n, k) {
  if (k <= 0 || k > n) {
    return(0)
  }

  if (n == 1 && k == 1) {
    return(1)
  }

  return(stirling_number(n - 1, k - 1) + (n - 1) * stirling_number(n - 1, k))
}

stirling_number_matrix <- function(n_max) {
  S <- matrix(0, nrow = n_max, ncol = n_max)
  S[1, 1] <- 1

  for (n in 2:n_max) {
    for (k in 1:n) {
      if (k == 1) {
        S[n, k] <- (n - 1) * S[n - 1, k]
      } else {
        S[n, k] <- S[n - 1, k - 1] + (n - 1) * S[n - 1, k]
      }
    }
  }
  return(S)
}

boswell_survival_func <- function(w, n) {
  # Compute the Stirling numbers of the first kind
  S <- stirling_number_matrix(n)
```

```r
  # Compute the survival function
  k_values <- 1:n
  total_sum <- sum((1 - pchisq(w, df = k_values + 1)) * S[n, k_values] / factorial(n))
  return(total_sum)
}
```

```r
# Define the critical value for the Laplace test
laplace_critical_value <- function(alpha) {
  return(qnorm(1 - alpha))
}

# Define the critical value for the Boswell test
boswell_critical_value <- function(alpha, n) {
  uniroot(function(w) boswell_survival_func(w, n) - alpha, lower = 0, upper = 100)$root
}
```

Boswell and Brunk (1969) observed empirically that the large sample approximation for the LRT yields a true significance level somewhat larger than the nominal level. We can empirically verify this observation in R using Monte Carlo simulations. The goal is to compare the true significance level of the LRT when using different nominal levels to see which one yields an actual significance level closer to 0.05.

```r
# Monte Carlo simulation to estimate the true significance level
monte_carlo_significance <- function(n, T_star, nominal_level, n_sim = 5000) {
  threshold <- boswell_critical_value(nominal_level, n)
  count_reject <- 0

  for (i in 1:n_sim) {
    # Simulate data under the null hypothesis (constant intensity)
    times <- simulate_hpp(n, T_star)

    # Perform Boswell's test
    test_stat <- boswell_test(times, T_star, n)

    # Check if the test statistic exceeds the threshold
    if (test_stat > threshold) {
      count_reject <- count_reject + 1
    }
  }
  # Estimate the true significance level
  true_significance <- count_reject / n_sim

  return(true_significance)
}
```

```r
# Parameters
n <- 20
T_star <- 2.0
n_sim <- 5000
nominal_levels <- c(0.05, 0.04)

# Run simulations for different nominal levels
results <- data.frame(
  Nominal_Level = nominal_levels,
```

```r
  True_Significance = sapply(nominal_levels, function(alpha) {
    monte_carlo_significance(n, T_star, alpha, n_sim)
  })
)

# Display the results
print(results)
```

```
##   Nominal_Level True_Significance
## 1          0.05            0.0662
## 2          0.04            0.0488
```

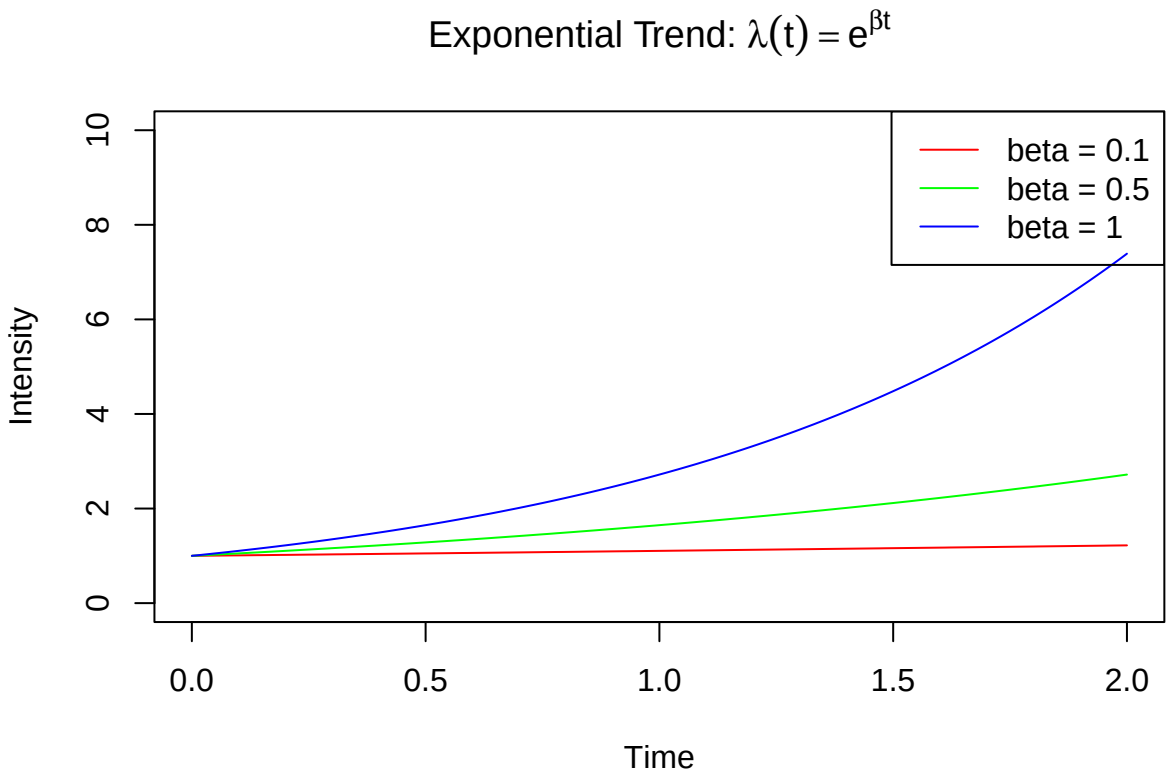
We can see that the true significance level is closer to the nominal level of 0.04 than 0.05.

Power Study

We conduct a power study to evaluate the performance of the Laplace and Boswell tests under different scenarios of increasing intensity functions.

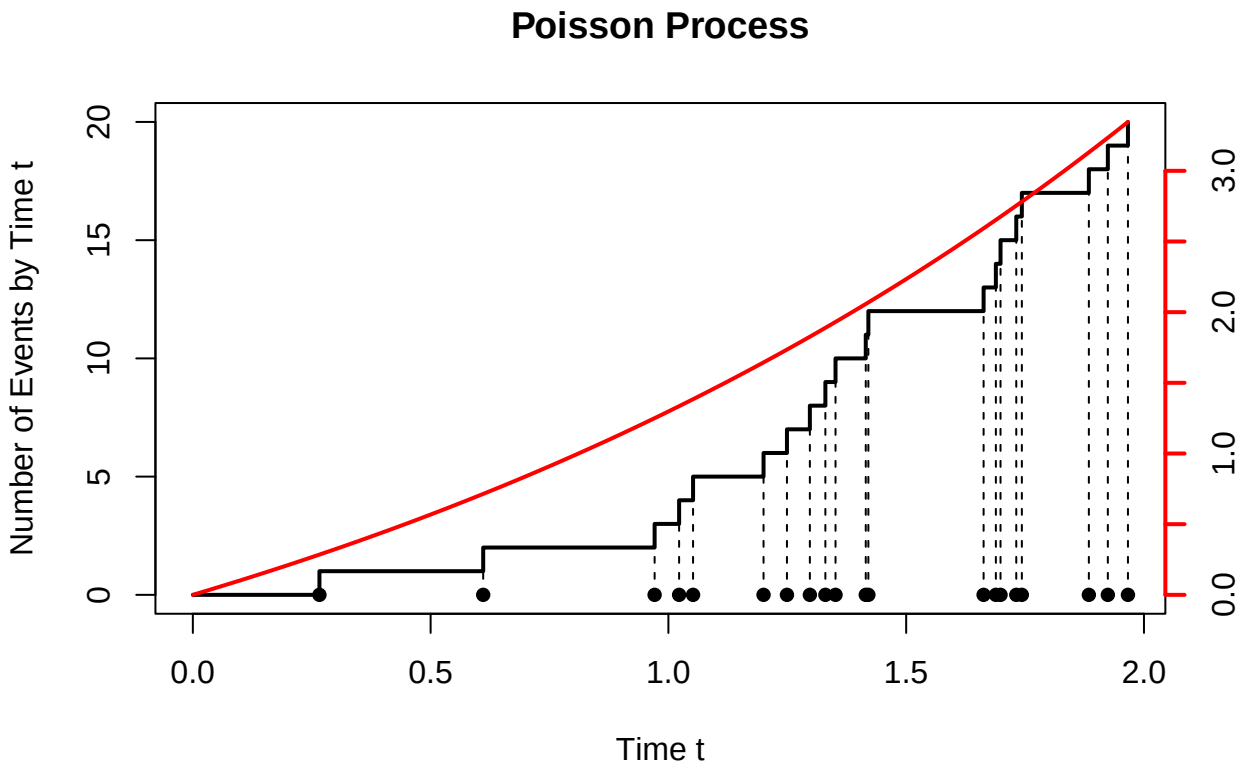
Exponential Trend

We first consider an exponential trend for different values of the parameter β.



We have simulated a Nonhomogeneous Poisson Process for each scenario (with $n = 20$, $T^* = 2.0$ and $\beta = 0.5$):

```r
# Simulation and plot of a Nonhomogeneous Poisson Process
n <- 20
T_star <- 2.0
beta <- 0.5
lambda_cum_func <- lambda_cum_exp(beta)
lambda_inv_func <- lambda_inv_exp(beta)

# Simulate NHPP data
times <- simulate_nhpp(n, T_star, lambda_cum_func, lambda_inv_func)

# Plot the NHPP data
plot.PP(times, lambda_cum_func)
```

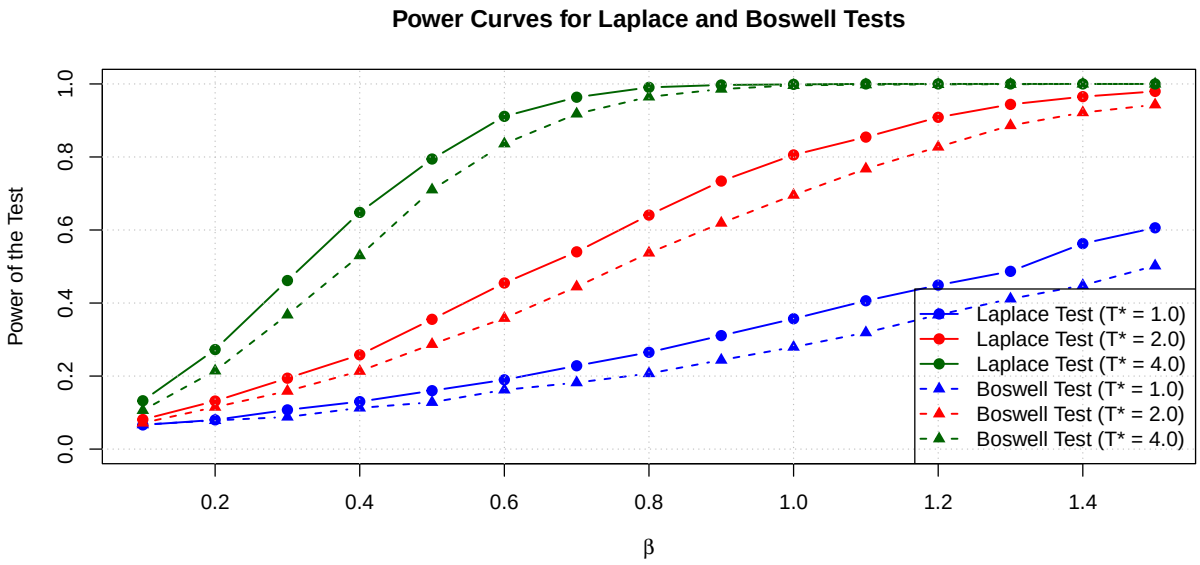


We then compute the power of the Laplace and Boswell tests for each scenario.

```r
# Compute the power of the Laplace and Boswell tests for each beta value
n <- 20
T_list <- c(1.0, 2.0, 4.0)
alpha.Laplace <- 0.05
alpha.Boswell <- 0.04
n_sim <- 5000
beta_values <- seq(0.1, 1.5, by = 0.1)

# Initialize lists to store power results
power_laplace <- list()
power_boswell <- list()
```

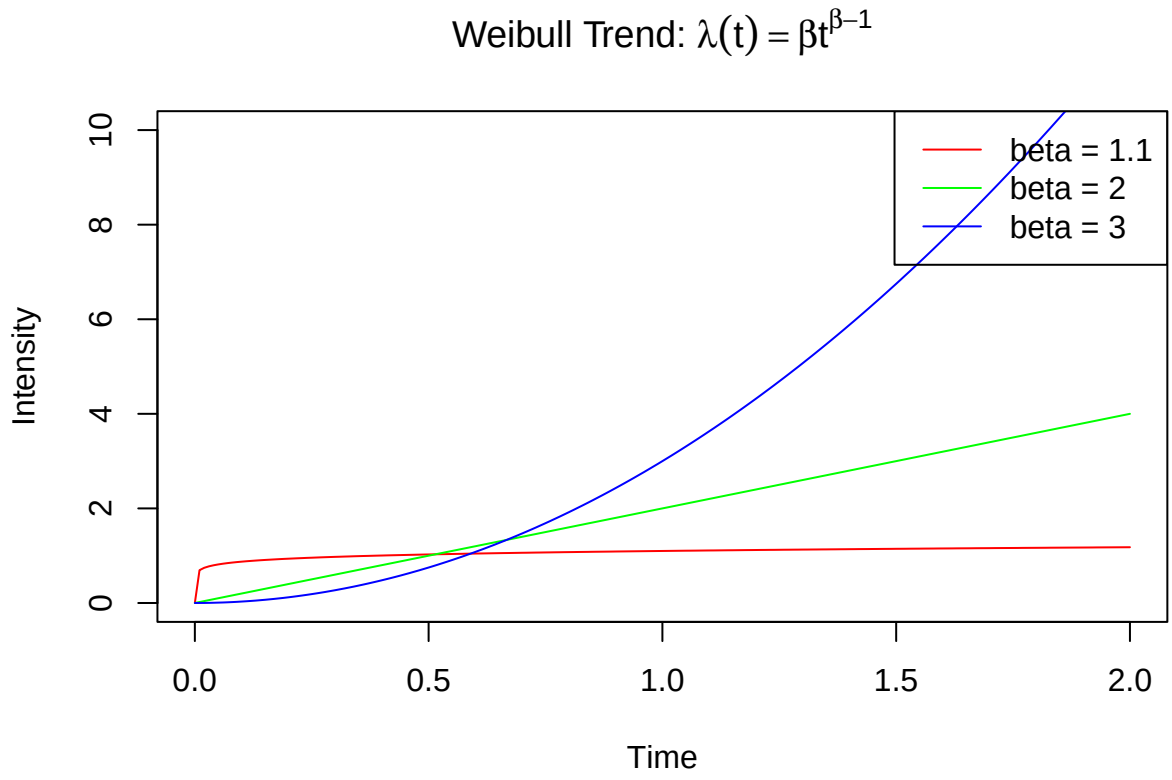
```r
# Compute the critical thresholds for the tests
threshold_laplace <- laplace_critical_value(alpha.Laplace)
threshold_boswell <- boswell_critical_value(alpha.Boswell, n)

for (T_star in T_list) {
  power_laplace_T <- numeric(length(beta_values))
  power_boswell_T <- numeric(length(beta_values))

  for (i in 1:length(beta_values)) {
    beta <- beta_values[i]
    lambda_cum_func <- lambda_cum_exp(beta)
    lambda_inv_func <- lambda_inv_exp(beta)

    # Compute power for the Laplace test
    power_laplace_T[i] <- monte_carlo_simulation(n, T_star, lambda_cum_func, lambda_inv_func,
                                                 laplace_test, threshold_laplace, n_sim)

    # Compute power for the Boswell test
    power_boswell_T[i] <- monte_carlo_simulation(n, T_star, lambda_cum_func, lambda_inv_func,
                                                  boswell_test, threshold_boswell, n_sim)
  }

  # Store the power results for the current T_star
  power_laplace[[as.character(T_star)]] <- power_laplace_T
  power_boswell[[as.character(T_star)]] <- power_boswell_T
}

# Combine power results into data frames for easier plotting
power_results_laplace <- data.frame(
  beta = rep(beta_values, length(T_list)),
  power = unlist(power_laplace),
  T_star = factor(rep(T_list, each = length(beta_values)))
)

power_results_boswell <- data.frame(
  beta = rep(beta_values, length(T_list)),
  power = unlist(power_boswell),
  T_star = factor(rep(T_list, each = length(beta_values)))
)
```

We plot the power curves for the Laplace and Boswell tests as a function of the parameter β with a significance level of $\alpha = 0.05$, $n = 20$, and $T^* = 1.0, 2.0, 4.0$.



Weibull Trend

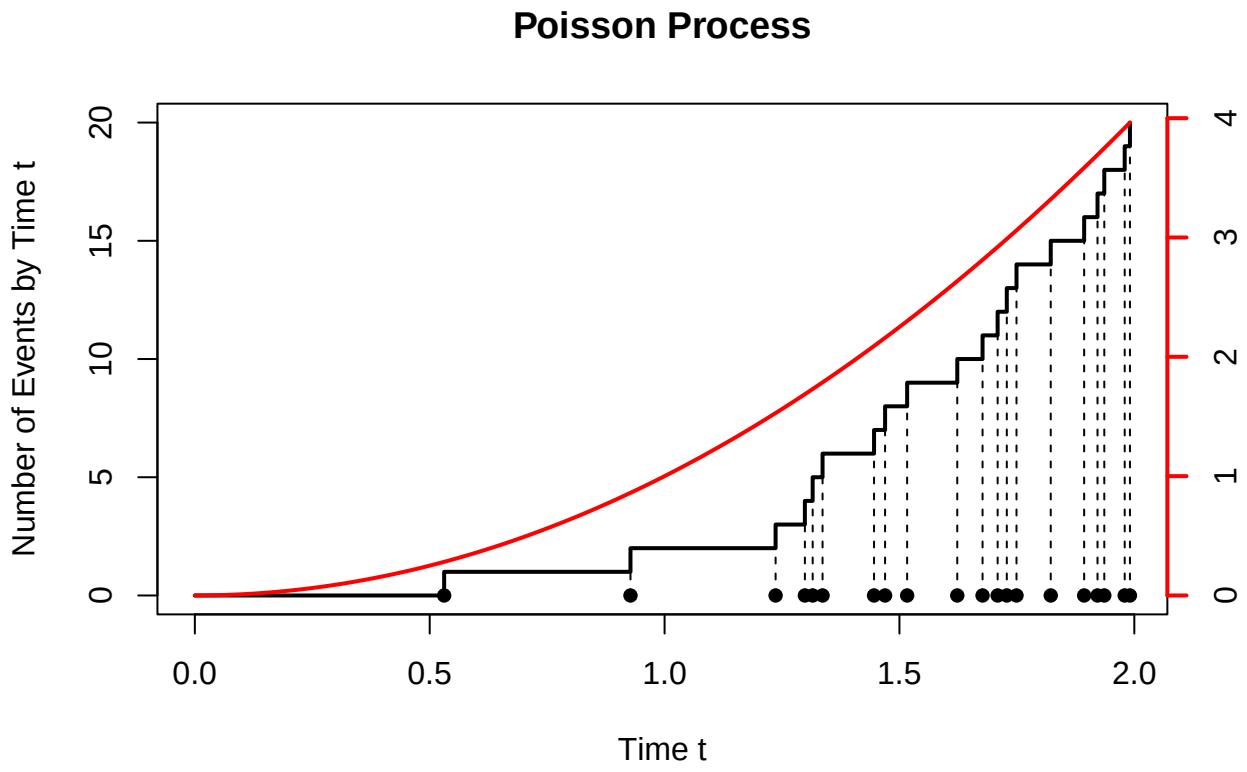
We now consider a Weibull trend for different values of the parameter β.



We have simulated a Nonhomogeneous Poisson Process for each scenario (with $n = 20$, $T^* = 2.0$ and

$\beta = 2.0$):

```
# Simulation and plot of a Nonhomogeneous Poisson Process
n <- 20
T_star <- 2.0
beta <- 2.0
lambda_cum_func <- lambda_cum_weibull(beta)
lambda_inv_func <- lambda_inv_weibull(beta)

# Simulate NHPP data
times <- simulate_nhpp(n, T_star, lambda_cum_func, lambda_inv_func)

# Plot the NHPP data
plot.PP(times, lambda_cum_func)
```

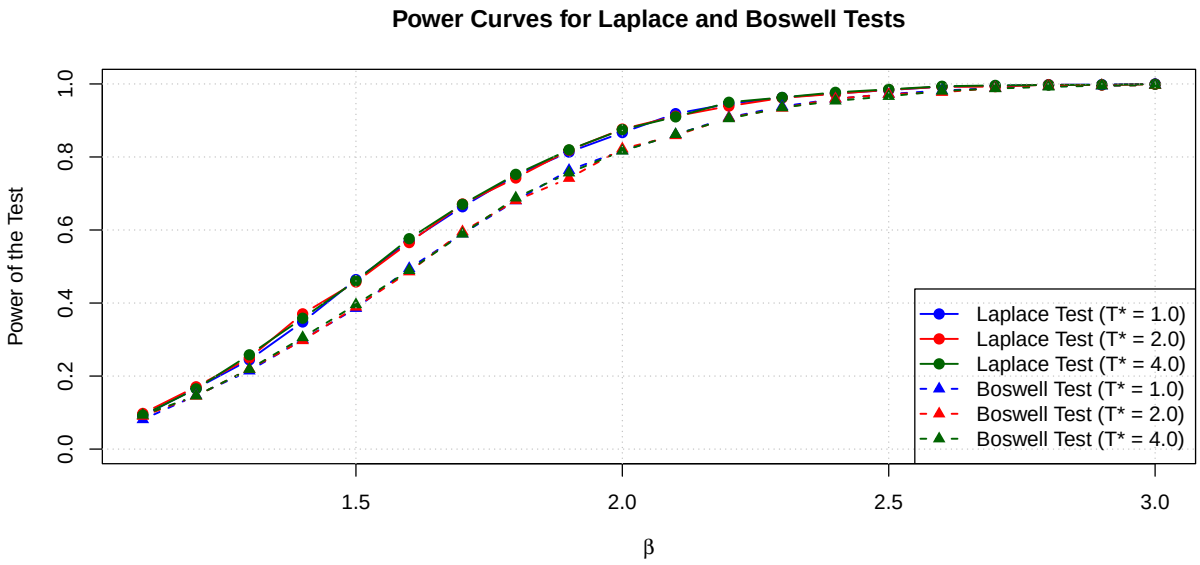


We then compute the power of the Laplace and Boswell tests for each scenario.

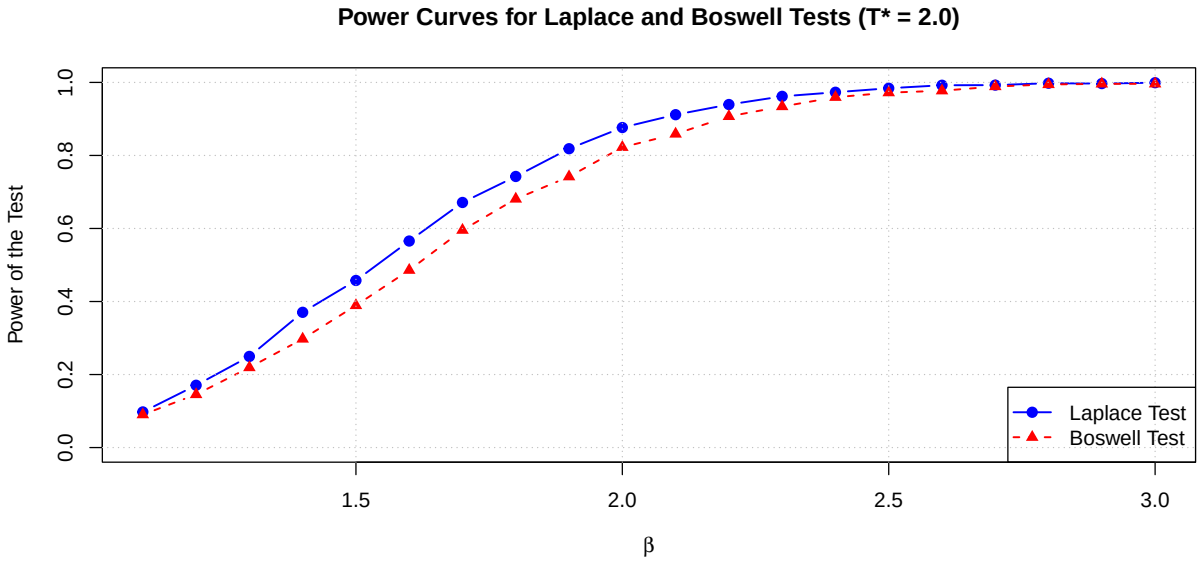
```
# Compute the power of the Laplace and Boswell tests for each beta value
n <- 20
T_list <- c(1.0, 2.0, 4.0)
alpha.Laplace <- 0.05
alpha.Boswell <- 0.04
n_sim <- 5000
beta_values <- seq(1.1, 3.0, by = 0.1)

# Initialize lists to store power results
```

```r
power_laplace <- list()
power_boswell <- list()

# Compute the critical thresholds for the tests
threshold_laplace <- laplace_critical_value(alpha.Laplace)
threshold_boswell <- boswell_critical_value(alpha.Boswell, n)

for (T_star in T_list) {
  power_laplace_T <- numeric(length(beta_values))
  power_boswell_T <- numeric(length(beta_values))

  for (i in 1:length(beta_values)) {
    beta <- beta_values[i]
    lambda_cum_func <- lambda_cum_weibull(beta)
    lambda_inv_func <- lambda_inv_weibull(beta)

    # Compute power for the Laplace test
    power_laplace_T[i] <- monte_carlo_simulation(n, T_star, lambda_cum_func, lambda_inv_func,
                                                 laplace_test, threshold_laplace, n_sim)

    # Compute power for the Boswell test
    power_boswell_T[i] <- monte_carlo_simulation(n, T_star, lambda_cum_func, lambda_inv_func,
                                                  boswell_test, threshold_boswell, n_sim)
  }

  # Store the power results for the current T_star
  power_laplace[[as.character(T_star)]] <- power_laplace_T
  power_boswell[[as.character(T_star)]] <- power_boswell_T
}

# Combine power results into data frames for easier plotting
power_results_laplace <- data.frame(
  beta = rep(beta_values, length(T_list)),
  power = unlist(power_laplace),
  T_star = factor(rep(T_list, each = length(beta_values)))
)

power_results_boswell <- data.frame(
  beta = rep(beta_values, length(T_list)),
  power = unlist(power_boswell),
  T_star = factor(rep(T_list, each = length(beta_values)))
)
```

We plot the power curves for the Laplace and Boswell tests as a function of the parameter β with a significance level of $\alpha = 0.05$, $n = 20$, and $T^* = 1.0, 2.0, 4.0$.



Since the power curves for the Weibull trend does not clearly show a difference between the time intervals $T^* = 1.0, 2.0, 4.0$, we will focus on the power curves for $T^* = 2.0$.



Application to Real-World Data

We apply the Laplace and Boswell tests to a real-world dataset to detect an increasing trend in the intensity of a Poisson process.

The dataset consists of the number of fire insurance claims in Denmark from 1980 to 1990. We are primarily interested in the arrival times of the claims (the time column) to detect if the rate of claims is increasing over time.

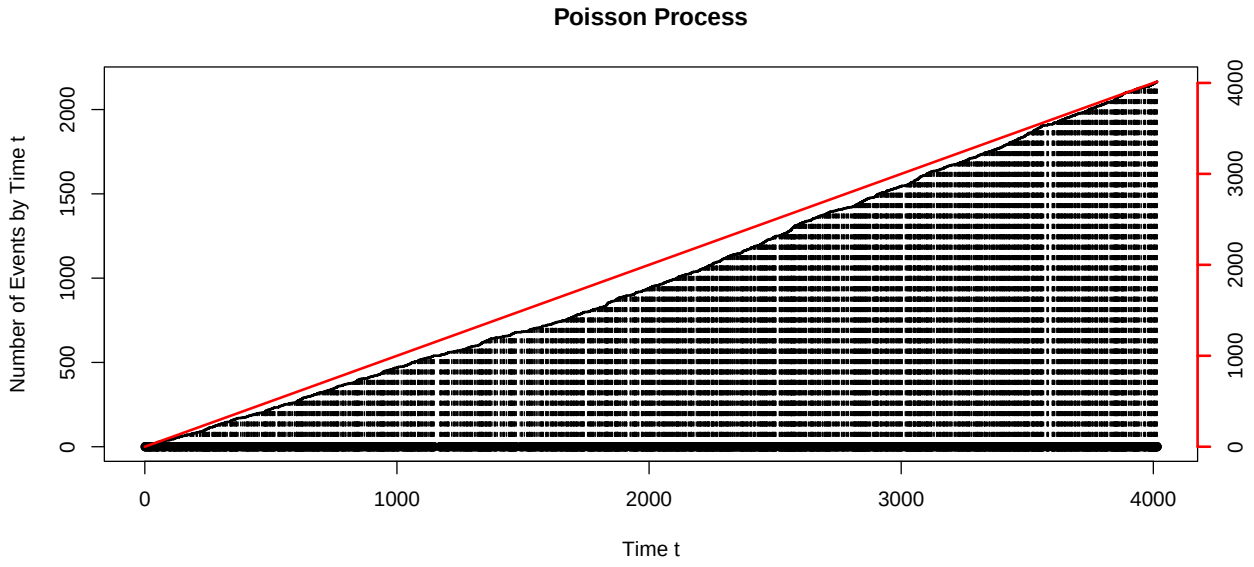
From this dataset, we were able to extract the times thanks to the library providing the database. Each

time corresponds to the date, day and time of the event. So we converted all these times into days, and then calculated a cumulative sum of the latter. For the moment, this allows us to display the start of the process, unchanged for the time being.

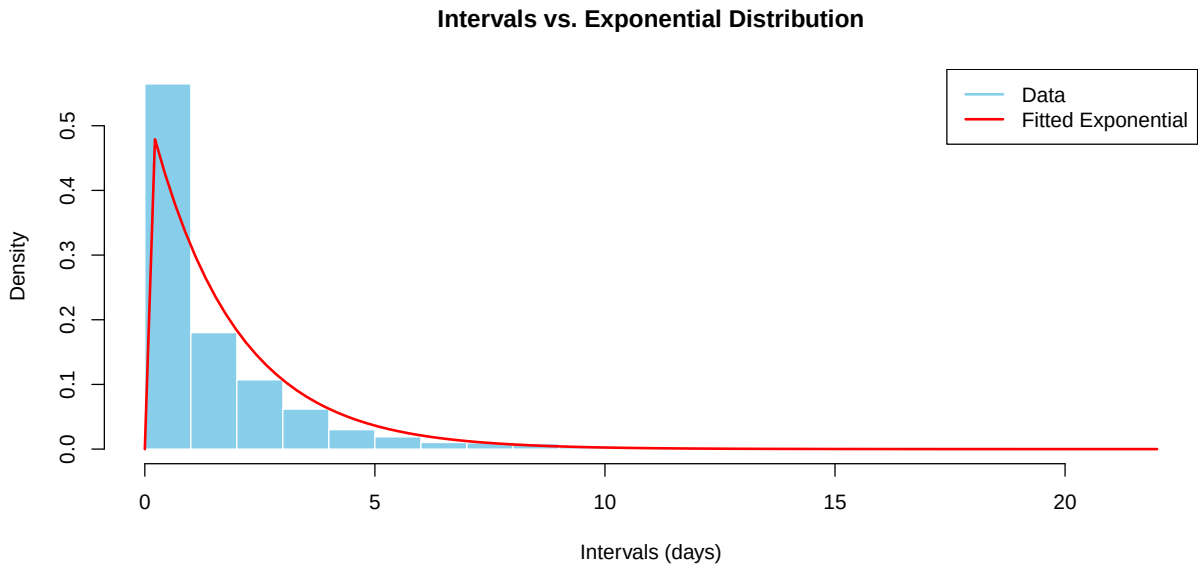
```
# Load the Danish dataset
data(danish)
df_data <- data.frame(Times = attr(danish, "times"))
times_intervals = as.numeric(diff(df_data$Times), units="days")

head(df_data)
```

```
##                 Times
## 1 1980-01-03 01:00:00
## 2 1980-01-04 01:00:00
## 3 1980-01-05 01:00:00
## 4 1980-01-07 01:00:00
## 5 1980-01-07 01:00:00
## 6 1980-01-10 01:00:00
```

We plot the cumulative number of claims over time to visualize the data.
We also plot the cumulative intensity function of a homogeneous Poisson process to compare it with the data.

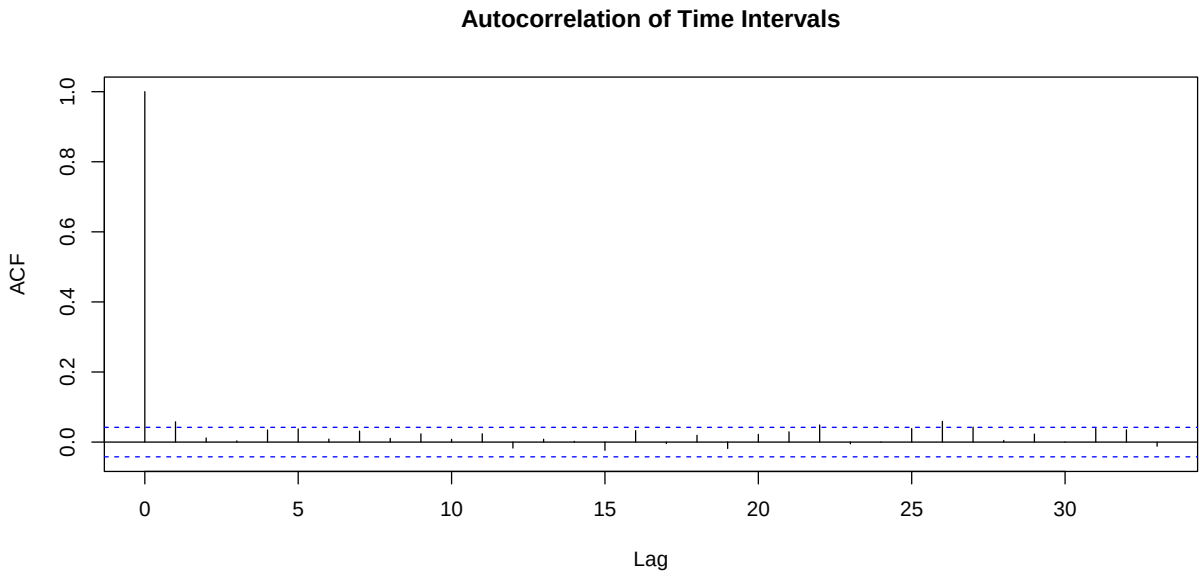


We adjust an exponential distribution to the time intervals to check if the inter-arrival times follow an exponential distribution.



We then check the autocorrelation of the time intervals to assess the independence of the time intervals.

```
# Autocorrelation of the time intervals
acf(times_intervals, main = "Autocorrelation of Time Intervals")
```



We ensure that the Poisson process is simple by removing duplicate time values.

```
# Remove duplicate time values
new_times_intervals <- times_intervals[times_intervals != 0]
```

```
# Variable declarations
times <- cumsum(new_times_intervals)
T_max <- max(times)
n <- length(times)
```

Application of the tests

We apply the Laplace and Boswell tests to the Danish dataset to detect an increasing trend in the intensity of the Poisson process.

Laplace Test

```r
# Laplace Test
Z_obs <- laplace_test(times, T_max, n)

# Compute the p-value
p_val <- 1 - pnorm(Z_obs)

print(p_val)
```

```
## [1] 0.0001150628
```

Boswell Test

```r
# Function to generate a matrix of log Stirling numbers of the first kind
stirling_number_log_matrix <- function(n_max) {
  log_S <- matrix(-Inf, nrow = n_max, ncol = n_max)  # Initialize with -Inf (log(0))
  log_S[1, 1] <- 0  # log(1) = 0, base case

  for (n in 2:n_max) {
    for (k in 1:n) {
      if (k == 1) {
        log_S[n, k] <- log_S[n - 1, n - 1] + log(n - 1)
      } else {
        log_S[n, k] <- log(exp(log_S[n - 1, k - 1]) + (n - 1) * exp(log_S[n - 1, k]))
      }
    }
  }
  return(log_S)
}

# Log factorial function
log_factorial <- function(n) {
  sum(log(1:n))
}

boswell_log_survival_func <- function(w, n) {
  # Ensure n does not exceed the precomputed matrix size
  log_stirling_matrix <- stirling_number_log_matrix(n)

  # Compute the log survival function
  total_sum <- 0

  for (k in 1:n) {
    log_term <- log(1 - pchisq(w, df = k + 1)) + log_stirling_matrix[n, k] - log_factorial(n)
```

```
    total_sum <- total_sum + exp(log_term)
  }

  return(total_sum)
}
```

```
# Boswell Test
W_obs <- boswell_test(times, T_max, n)

# Compute the p-value
p_val <- boswell_log_survival_func(W_obs, n)

print(p_val)
```

We plot the value of w_α as a function of n to check if we can find an approximate value for the critical value.

```
# Plot the critical value for the Boswell test
n_values <- seq(10, 170, by = 10)
alpha <- 0.04
w_alpha_values <- sapply(n_values, function(n) boswell_critical_value(alpha, n))

plot(n_values, w_alpha_values, type = "b", col = "blue", lwd = 2,
     xlab = "n", ylab = expression(w[alpha]),
     main = "Critical Value for the Boswell Test")
```