Report II

Periodic inspections and interval censored data

Petronela Pawlisz (230032), Patryk Wielopolski (234891)

Introduction

In this report we will focus on periodic inspections and interval censored data. We will consider simple situation of lightbulb which is periodicaly inspected if it has failed. This quite simple example will enable us to understand censoring mechanism and some interesting properties of such kind of processes. Moreover we will get familiar with estimation methods of survival function.

Generator

Our implementation of generator is presented on Listing 1. Firsly we create counters for current time, last time of lightbulb checking and last time of lightbulb changing. Also we initialize vectors and list which will store our results. Implementation of event loop is a bit straightforward. We are going through while loop till the current time is smaller than simulation end time. Then we check if lightbulb failed. If yes, then we save intervals, change lightbulb to new one and generate its time of failure and save that time. After that we generate new inspection time. When the while loop ends, we add last observations, which wasn't censured. The function returns list of inspection times, light failure times, censored intervals and initial parameters.

```
# Lambda - failure rate
# Nu - inspection rate

generate_censored_data <- function(lambda, nu, time_end) {
  time_now <- 0
  lightbulb_next_failure <- rexp(1, rate = 1/lambda)

  inspection_times <- c()
  light_failures_times <- c(lightbulb_next_failure)
  intervals <- list()

  lightbulb_last_check <- 0
  lightbulb_last_change <- 0

  while (time_now < time_end){
    if (time_now > lightbulb_next_failure){
      # Save censored interval of failure
      intervals$left <- c(intervals$left, lightbulb_last_check - lightbulb_last_change)
      intervals$right <- c(intervals$right, time_now - lightbulb_last_change)
      intervals$censored <- c(intervals$censored, 1)
      # Change lightbulb and generate next failure time
      lightbulb_last_change <- time_now
      lightbulb_next_failure <- time_now + rexp(1, rate = 1/lambda)
```

```
      # Save real time of future failure
      light_failures_times <- c(light_failures_times, lightbulb_next_failure)
    }
    lightbulb_last_check <- time_now
    inspection_times <- c(inspection_times, lightbulb_last_check)
    time_now <- time_now + rexp(1, rate = 1/nu)
  }

  intervals$left <- c(intervals$left, lightbulb_last_check - lightbulb_last_change)
  intervals$right <- c(intervals$right, Inf)
  intervals$censored <- c(intervals$censored, 0)

  return(list(
    inspection_times=inspection_times,
    light_failures_times=light_failures_times,
    intervals=intervals,
    lambda=lambda,
    nu=nu,
    time_end=time_end))
}
```

Listing 1: Implementation of generator.

From theoretical aspect it's worth to mentions that process of inspections is a Poisson process because it starts in 0, its increases are independent and its waiting times for next event are independent and have exponential distribution. Besides process of lightbulb changes is not a Poisson process because waiting times are not idependet.

Sample realisations of process can be found on figure 1. There are two examples which shows times when inspection has occured (black dots) and lightbulb has died (red dots). As we can observe everything looks correct, especially failures and inspections are in correct order. But to be sure we will make further analysis in next section.

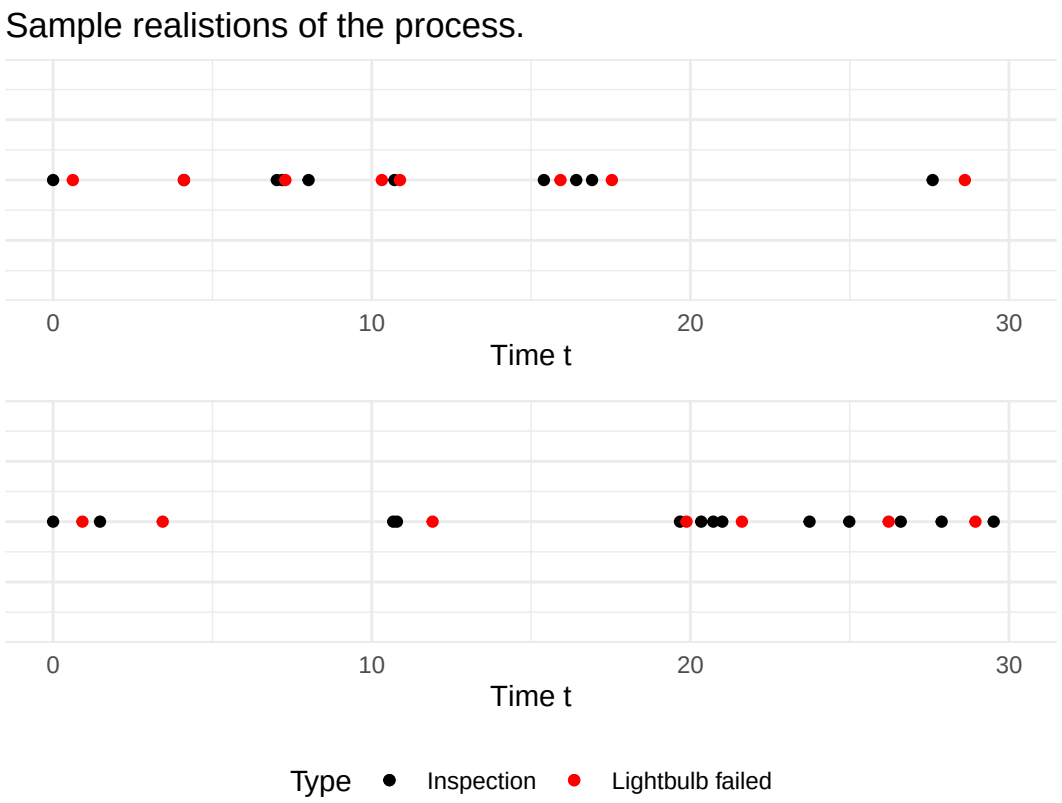

Figure 1: Sample realisation of the process.

Analysis of generator

Before we start estimations of survival function we will take a closer look to our generator. To do that we have simulated 300 times our process for grid of parameters:

- λ - $[1, 10]$ with step equal 1,
- ν - $[1, 10]$ with step equal 1,
- T_0 - $[100, 700]$ with step equal 300.

We want to check following statistics:

- number of lightbulb replacements,
- percentage of time without light,
- number of lightbulb inspections,
- number of lightbulb failures,
- percentage of inspections where left interval was non-zero.

Our results are presented on plots with following structure:

- x-axis contains λ - lightbulb failure rate,
- y-axis contains value of statistics,

- color represents ν - inspection rate,
- grid represents T_0 value.

Figure 2 represents number of lightbulb replacements. Our intuitions is that low rate of ν which corresponds to frequent checking of lightbulb will give us the highest value of lightbulb replacements. When we get higher value of this parameter, we will get smaller value of statistic. Also to our minds taking higher value of λ will corresponds to smaller value of replacements because lightbulb will be more durable. As we can see on the plot all our intuisions are similar to obtained results. Also it's worth to mention that parameter T_0 only scales our plot but doesn't affect shape of curves.

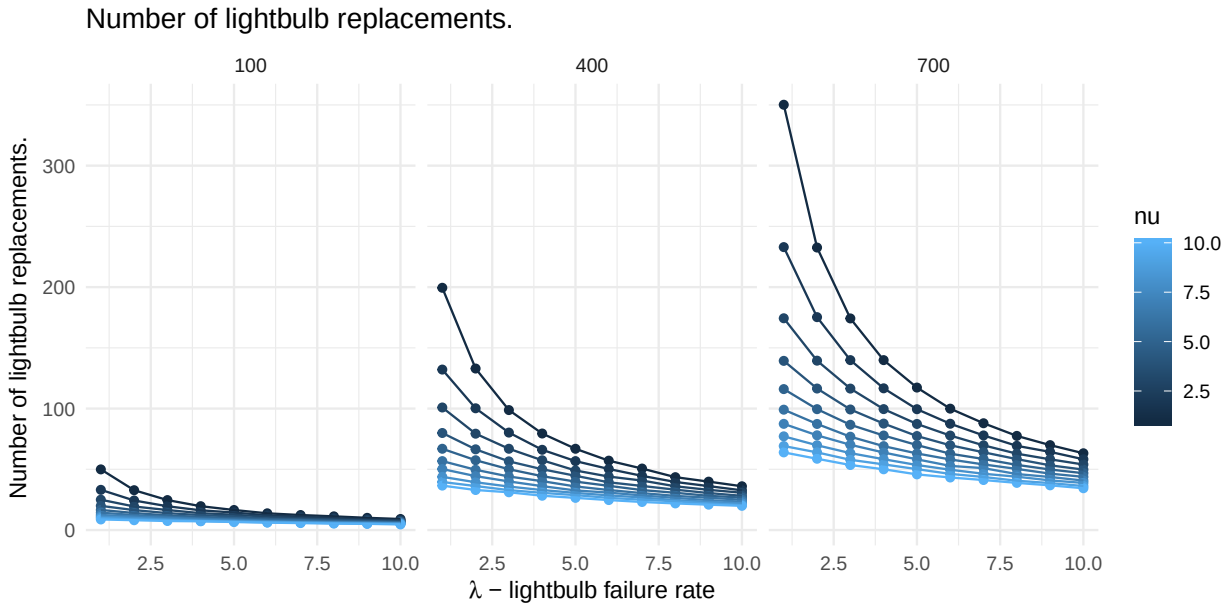

Figure 2: Number of lightbulb replacements in setup of different parameters of process. Color represents value of ν - inspection rate.

The second plot, which we will be analysing and can be found on Figure 3, corresponds to statistic 'percentage of time without light'. It may be very crucial information from application perspective because from this plot we could deduce how often we should check our lightbulb to obtain specific level of percentage of time without light (and going beyond that this answears the question why we care about estimation methods of survival functions and parameter λ). Our intuintions for that plot is that for high value of λ and small value of ν we will get the smallest value of statistic and conversely for small value of λ and high value of ν we will get the highest value of statistic. First situation corresponds to very frequent checking of lightbuld which is durable and second one to very rare checking of not durable lightbulb. Our intuisions match to obtained results. Also parameter T_0 has no bigger influence to simulations.

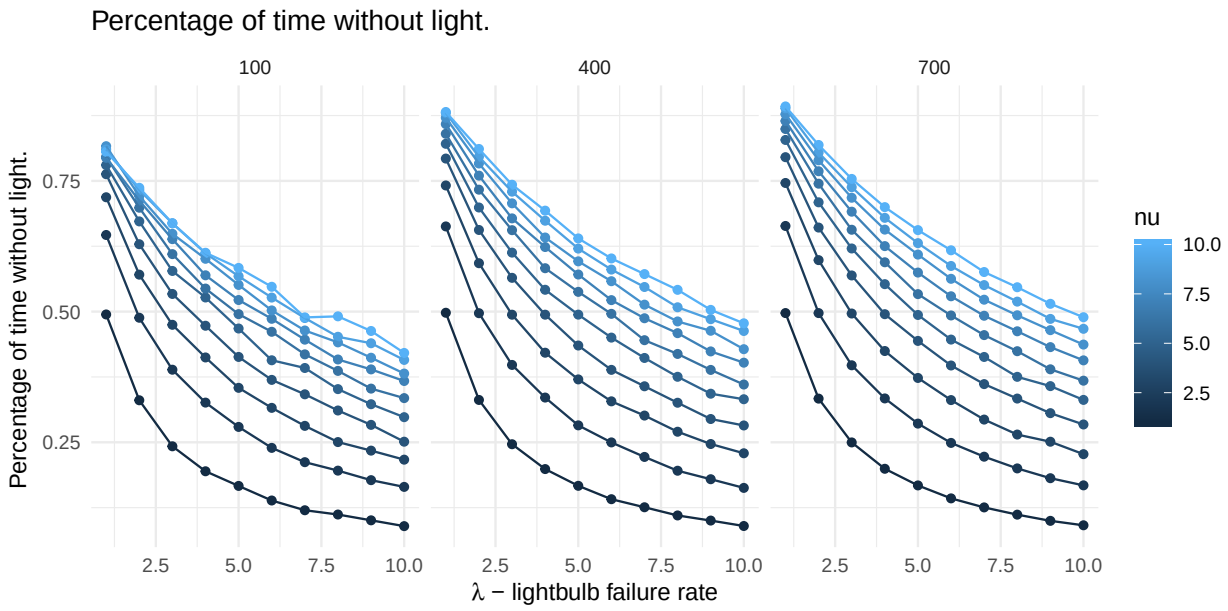

Figure 3: Percentage of time without light in setup of different parameters of process. Color represents value of ν - inspection rate.

The next plot represents number of lightublb inspections and can be found on figure 4. Intuitions for that plot are quite straightforward. Inspections are Poisson process so we should obtain number of lightbulbs equal to $\frac{T_0}{\nu}$. Indeed our simulations confirms that and also it can be treated as empirical proof of previous fact. Also it's worth to mentation that these statistics doesn't depend on process of lightbulb failures.

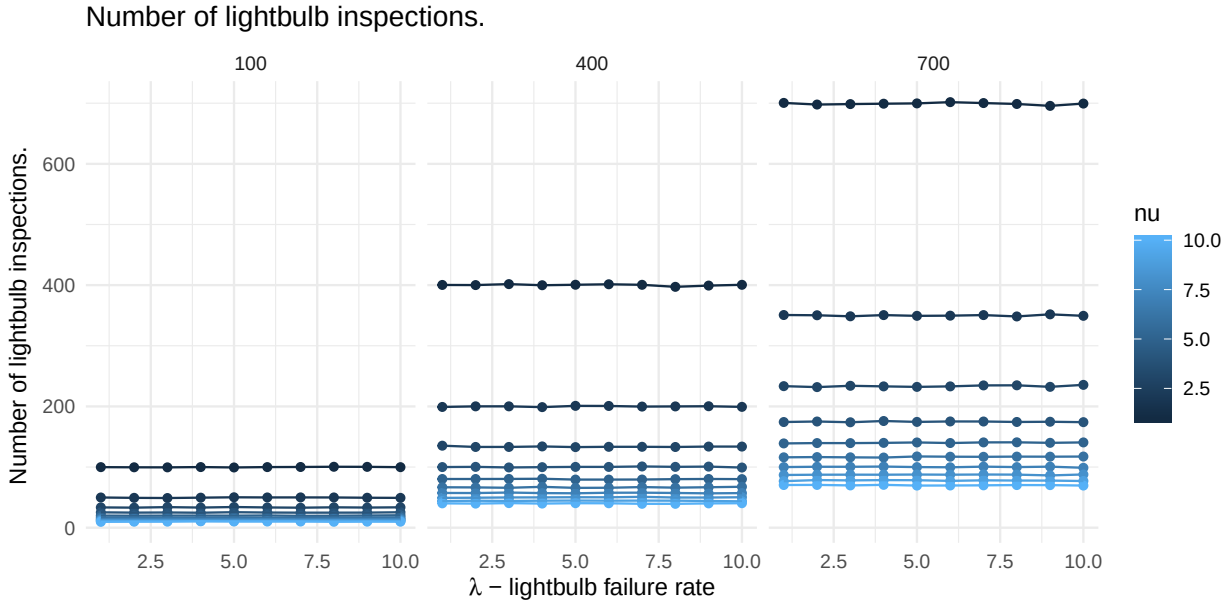

Figure 4: Number of lightbulb inspections in setup of different parameters of process. Color represents value of ν - inspection rate.

The next plot is presented on Figure 5. It represents value of number of lightbulb failures. If we think about

our process we can conclude that the result highly depends on ν parameter because it's responsible for how often we detect that lightbulb has failed. So our intiusions are that as ν will be smaller the more failures we will observe and higher values of λ will provide us smaller number of failures because then lightbulbs will be more durable. As we can observe on figure, our thougts are correct and it confirms our belifes that our generator is performing in correct way.

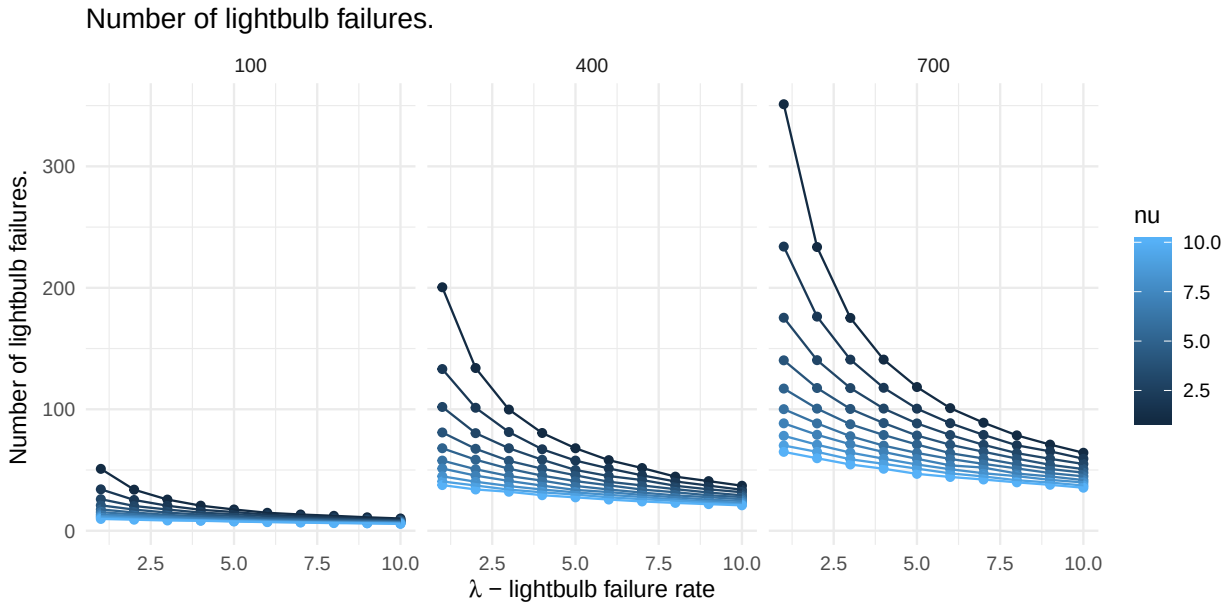

Figure 5: Number of lightbulb failures in setup of different parameters of process. Color represents value of ν - inspection rate.

The last one plot in this section is presented on Figure 6. It represents statistic 'percentage of inspections where left interval was non-zero' which could be interpreted what percentage of observations has more than one inspection before failure which gives us better observation to our estimator. Our intuitions are that small ν and big λ will give us the best (small) value because we have frequent checks and rare failures. Conversly high value of ν and small value of λ will give us the worst results because we have frequent failures and check that rarely. Simulations confirm our suppositions. It's worth to mention that we can observe how the curves are getting smoother in increase of T_0. It's probably due to fact that as we have longer simulations we have more observations and estimations of mean value of statistics are better.

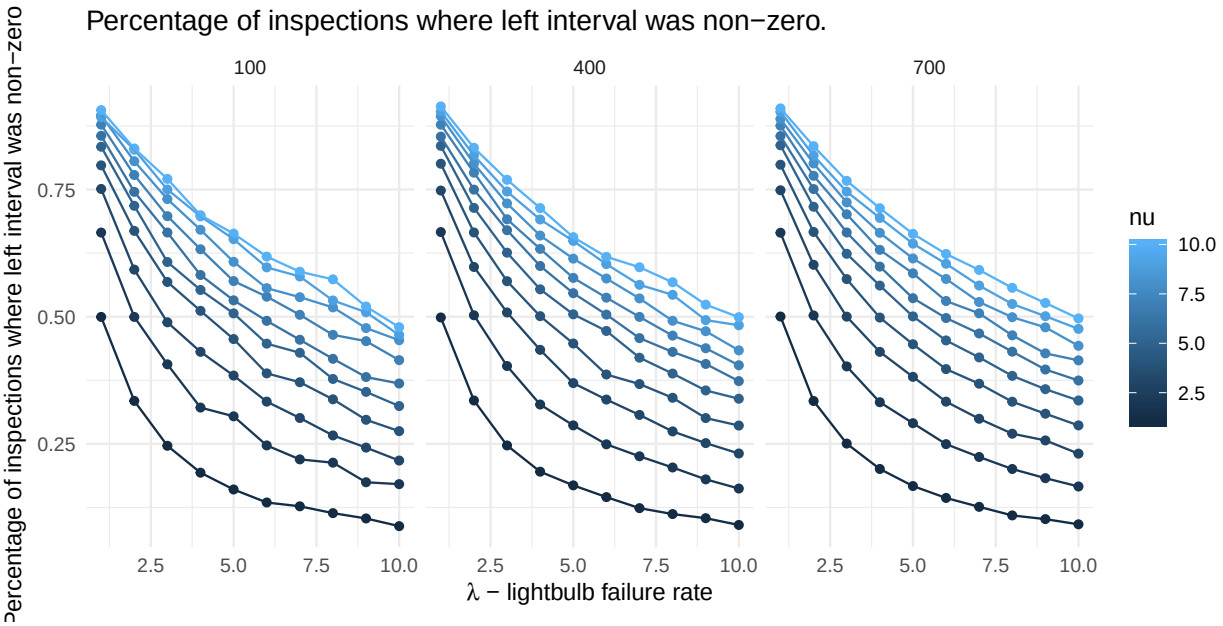

Figure 6: Percentage of inspections where left interval was non-zero in setup of different parameters of process. Color represents value of ν - inspection rate.

In this part we have analyzed some interesting statistics which enabled us to better understand considered process and confirm that our generator works properly. After this analysis we can smoothly go to estimation problem.

Naive estimator

We want to estimate the failure rate using an average of finite right sides of intervals. This is a naive estimator which ignores the censoring. We want to check if how this estimator depends on choosen ν and λ.

Mean of the estimator

Our intuitions is that the estimated values will be higher than the real values. By taking the time of lightbub checking as the time of lightbulb fail we add to the lifetime of lightbulb some extra time. The time of lightbulb work is artificially elongated, which results in higher values of λ estimator.

For bigger of ν frequency of checking lightbulb is smaller which can results in longer time between lightbulb fail and lightbulb replacemet than for more freqent checking. We expect that the bigger ν, the bigger difference bewteen estimated and the real value will be.

At figure 7 we observe the estimated values of the failure rate and the real values of λ (red, dotted line). As we predicted the estimated values are higher than the real ones and the bigger value of ν, the bigger difference bewteen estimated and the real value is.

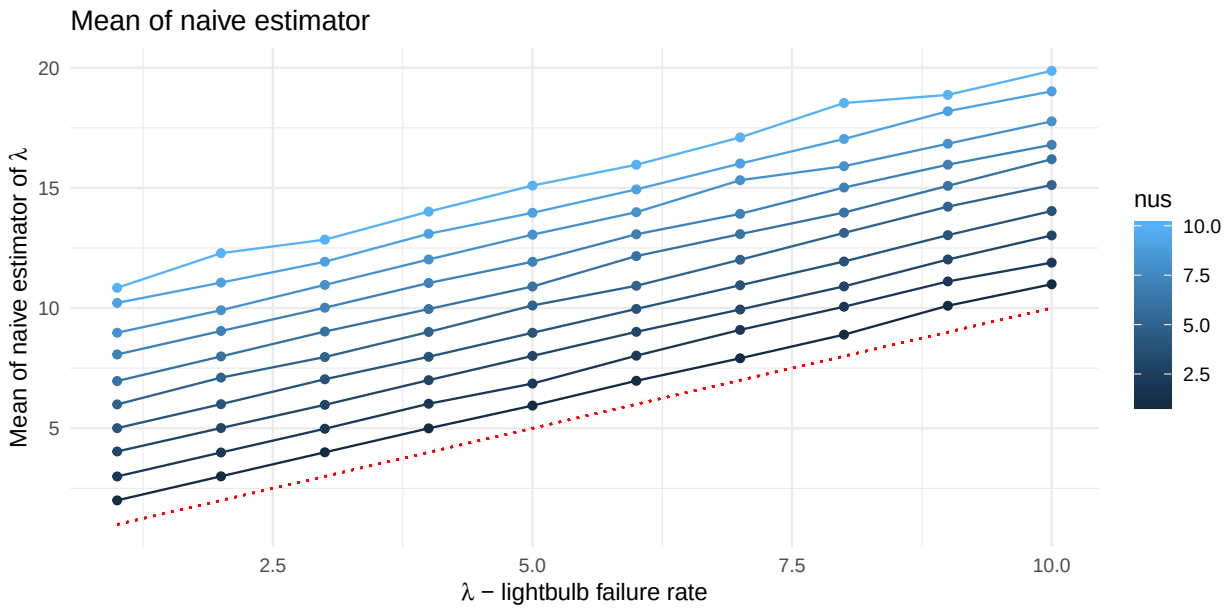

Figure 7: Mean of naive estimator of the failure rate.

Variance of the estimator

The low rate of ν which corresponds to frequent checking of lightbulb which should result in less time bewteen lightbulb fail and lightbulb replacement. The low rate of λ corresponds to highest value of lightbulb replacements and as with small ν it should reduce the time between lightbulb fail and lightbulb replacement. Due to that our intuition is that the estimated values will be closer to the real ones for smaller values of ν and λ. We expect that with ν and λ growth variance of naive estimator will also grow.

At figure 8 we observe that the variance of naive estimator is bigger for bigger ν and λ values as we predicted. The growth is exponential.

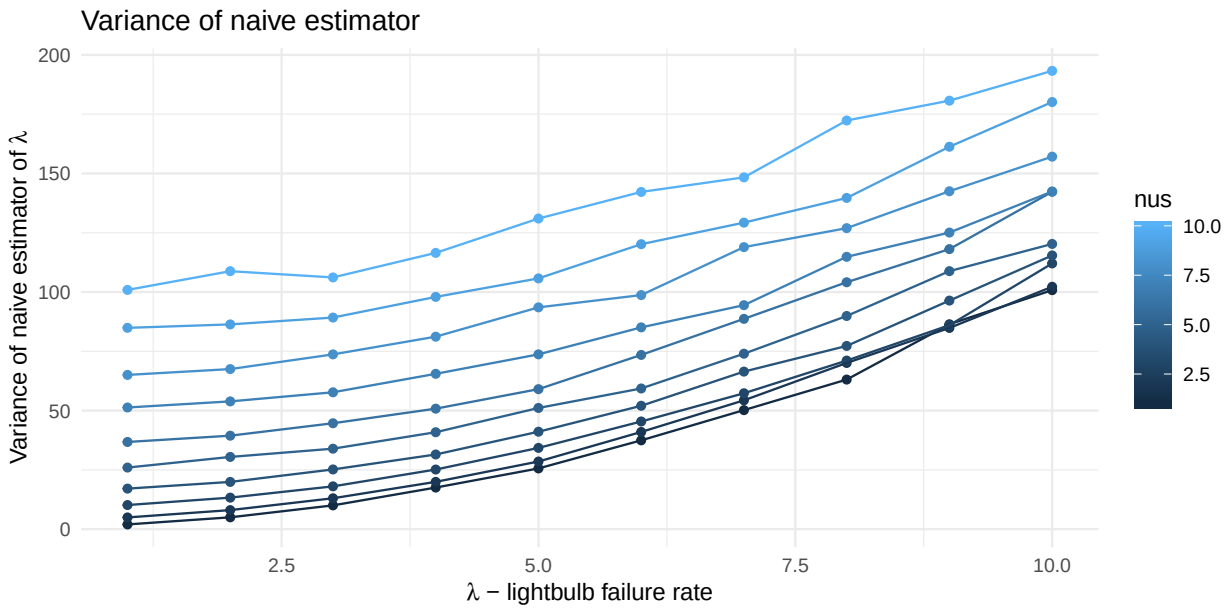

Figure 8: Variance of naive estimator of the failure rate.

Bias of the estimator

Based on previous observations we expect that with ν growth the bias of naive estimator will also grow.

At figure 9 we observe that the bias of naive estimator is bigger for bigger ν values as we predicted. For one chosen ν bias is constant (do not depends on λ).

This estimator is not unbaised. Unbaised estimator is when bias is equal 0 (red, dotted line).

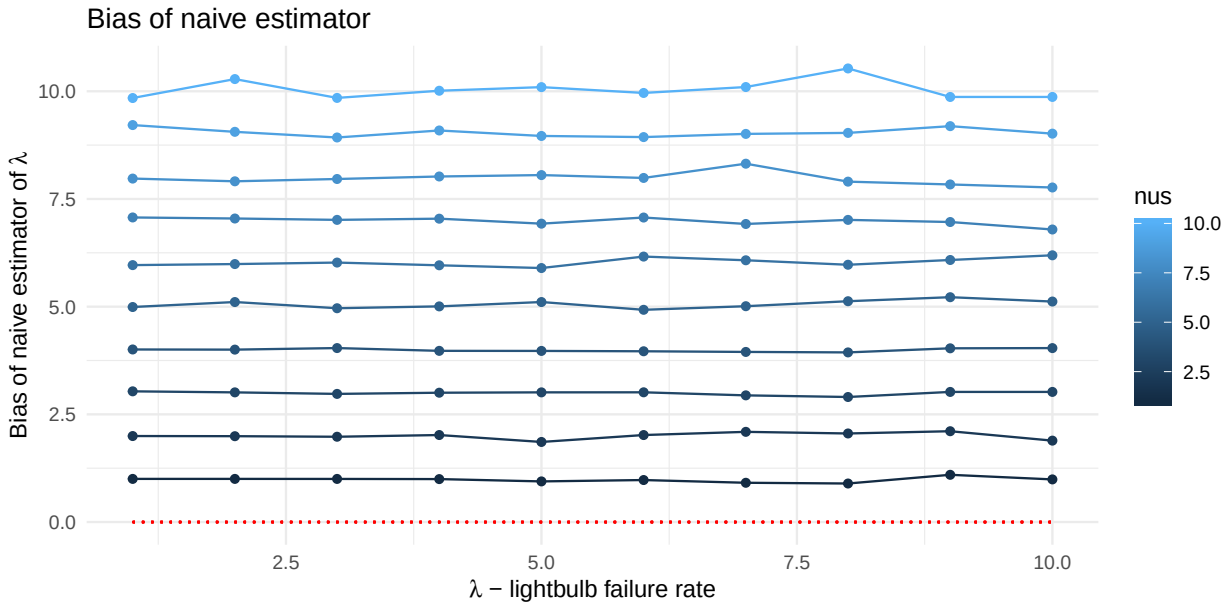

Figure 9: Bias of naive estimator of the failure rate.

Mean square error of the estimator

As previously our intuition is that the estimated values will be closer to the real ones for smaller values of ν and λ, so the mean square error should be the smallest for the smallest chosen ν and λ.

At figure 10 we observe that in fact the mean square error of naive estimator is bigger for bigger ν and λ values as we predicted. It is the exponential growth.

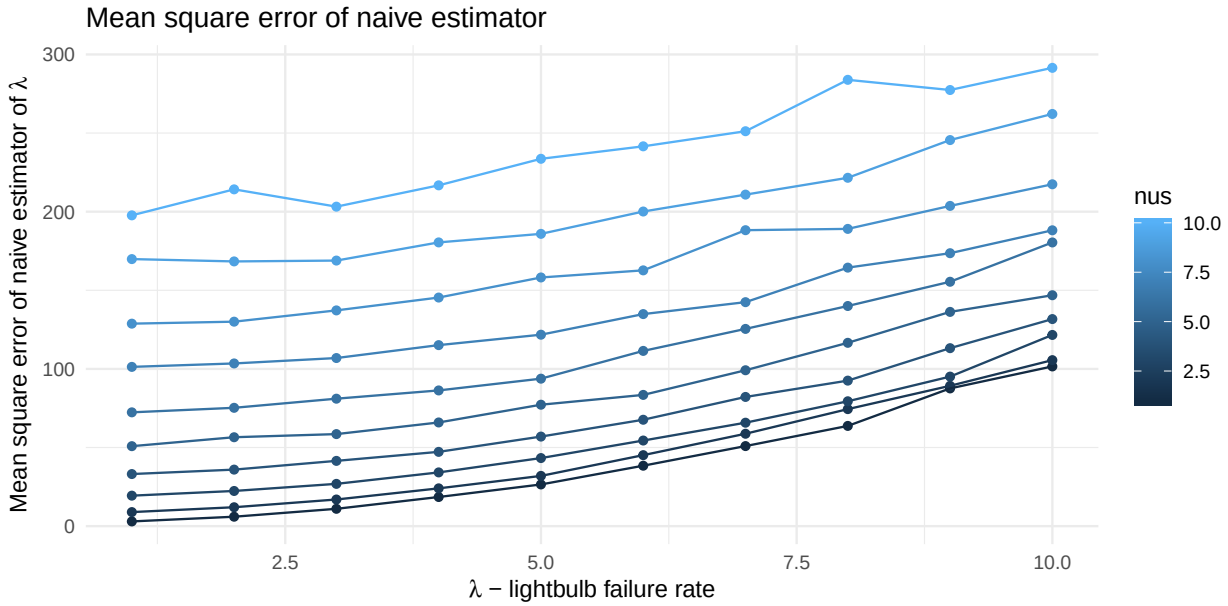

Figure 10: Mean square error of naive estimator of the failure rate.

Taking into account interval censoring

We created our own implementation of Turnbull estimator of survival function based on notes from lecture. Our implementation of this procedure is presented on Listing 2.

```
source('src/generator.R')
source('src/plots.R')

turnbull <- function(lambda, nu) {
data <- generate_censored_data(lambda, nu, 500)
vector_intervals <- c(data$intervals$left, data$intervals$right)

E <- matrix(0,length(vector_intervals),2)
count <- seq(length(vector_intervals))
for (i in count){
  E[i,1] <- vector_intervals[i]
  if (i < length(data$intervals$left)+1){
    E[i,2] <- 2
  } else {
    E[i,2] <- 1
  }
}
E[i,2] <- -2
```

```
E_2 <- E[order(E[ ,1]), ]
loop_num <- seq(length(vector_intervals)-1)
for (j in loop_num){
  if (E_2[j,1] == E_2[j+1,1] & E_2[j,2] > E_2[j+1,2]){
    a <- E_2[j, ]
    b <- E_2[j+1, ]
    E_2[j, ] <- b
    E_2[j+1, ] <- a
  }
}

intervals_left <- c()
intervals_right <- c()
type_left <- c()
type_right <- c()
for (k in loop_num){
  if (E_2[k,2] == 2 & E_2[k+1,2] == 1 || E_2[k,2] == 2 & E_2[k+1,2] == -2 ){
    intervals_left <- c(intervals_left, E_2[k,1])
    type_left <- c(type_left, E_2[k,2])
    intervals_right <- c(intervals_right, E_2[k+1,1])
    type_right <- c(type_right, E_2[k+1,2])
  }
}

org_intervals <- data.frame(E[1:length(data$intervals$left), ],
                            E[length(data$intervals$left) + 1 : length(data$intervals$left
turnbull_interval <- data.frame(intervals_left, type_left, intervals_right, type_right)


n <- length(data$intervals$left)
m <- length(intervals_left)

A <- matrix(0,n,m)

for (k in 1: n){
  for (l in 1:m){
    if (turnbull_interval[l,1] >= org_intervals[k,1] &
        turnbull_interval[l,3] <= org_intervals[k,3]){
      A[k,l] <- 1
    }
  }
}


count_interval_values <- function(A, m, eps=0.001, max_steps=100) {
s <- rep(1/m, m)

  for (i in 1:max_steps) {
    d_j <- colSums(A*s/rowSums(A*s))
    n_j <- rev(cumsum(rev(d_j)))
    p_j <- (n_j - d_j)/ n_j
    S_j <- cumprod(p_j)
    s_temp <- c(1, S_j[1:length(S_j)-1]) - S_j
```

```
    if (sum(abs(s-s_temp)) < eps) {
      break
    }
    s <- s_temp
  }
  return(s)
}

prob <- count_interval_values(A, m)


length_intervals_decrease <- turnbull_interval[ ,3]- turnbull_interval[ ,1]
length_intervals_constant <- turnbull_interval[ ,1] - c(0, turnbull_interval[1:m-1 ,3])


heights <- 1
for (j in 1:m){
  heights <- c(heights, heights[j]-prob[j])
}

estimated_mean <- sum(length_intervals_constant*heights[1:m] + length_intervals_decrease*(
return(estimated_mean)
}
```

Listing 2: Implementation of generator.

Simply speaking Turnbull procedure orders the left and right end-points on single time-line and picks intervals for which right end-point follows immediately after left endpoint. The exemplary results of this procedure we can see in Table 1.

Table 1: Generated intervals (first table) and intervals after Turnbull procedure (second table) - example

intervals_left	type_left_left	intervals_right	type_right
0.9733396	2	6.069197	1
0.0000000	2	1.134347	1
0.0096717	2	2.952830	1
2.2626930	2	4.518257	1
6.8713983	2	7.861387	1
8.1129480	2	8.554828	1
0.0000000	2	3.304613	1
0.0000000	2	1.962345	1
0.0000000	2	3.437338	1
0.1834686	2	3.786097	1
0.0000000	2	1.152759	1
0.7484737	2	Inf	-2

intervals_left	type_left	intervals_right	type_right
0.9733396	2	1.134347	1
2.2626930	2	2.952830	1
6.8713983	2	7.861387	1
8.1129480	2	8.554828	1

To estimate the failure rate we integrate area under the survival function.

We want to check if how this estimator depends on choosen ν and λ.

Mean of the estimator

Our intuitions is that Turnbull estimator will give us better (closer to the real ones) results than the naive one.

At figure 11 we observe the estimated values of the failure rate and the real values of λ (red, dotted line). As we predicted the estimated values are smaller than for the naive estimator (figure 7), and the bigger value of ν, the bigger difference bewteen estimated and the real value is.

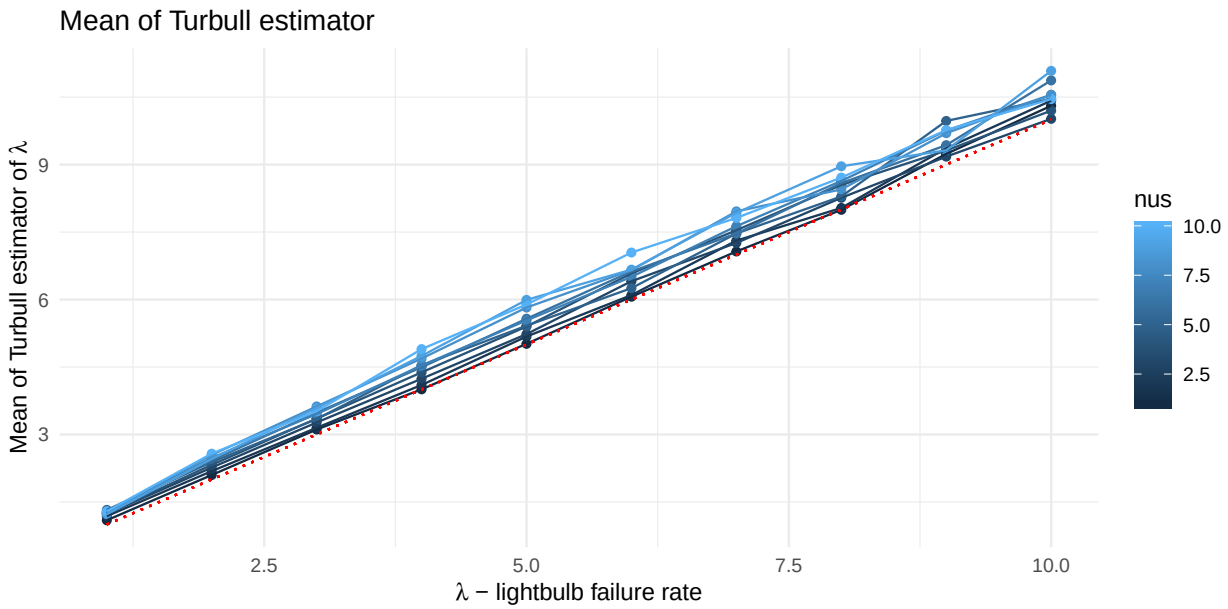

Figure 11: Mean of Turnbull estimator of the failure rate.

Variance of the estimator

As we predicted the variance of estimated values (figure 12) are much smaller than for the naive estimator (figure 8), and as previously the variance of naive estimator is bigger for bigger ν and λ values. The curve is not as smooth as for the naive estimator.

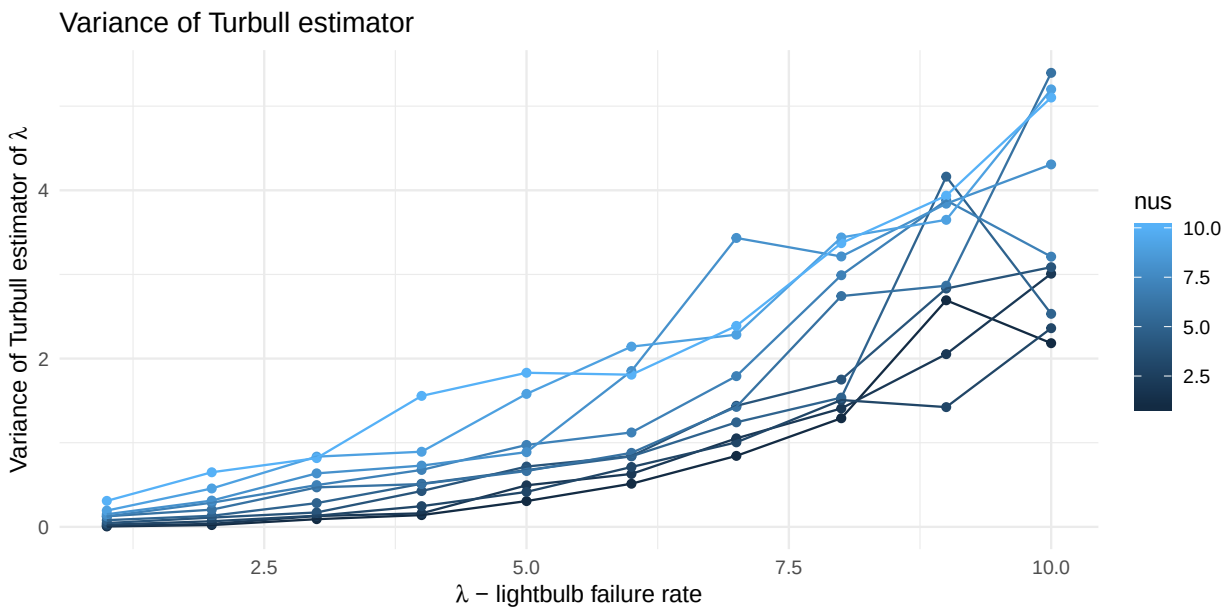

Figure 12: Variance of Turnbull estimator of the failure rate.

Bias of the estimator

As we predicted the bias values of Turnbull estimator (figure 13) are smaller than for the naive estimator (figure 9). This time the bias depends not only on ν (as it was with naive estimator) but changes also with change of λ. For one chosen ν bias is not constant.

This estimator is also not unbaised. Unbaised estimator is when bias is equal 0 (red, dotted line).

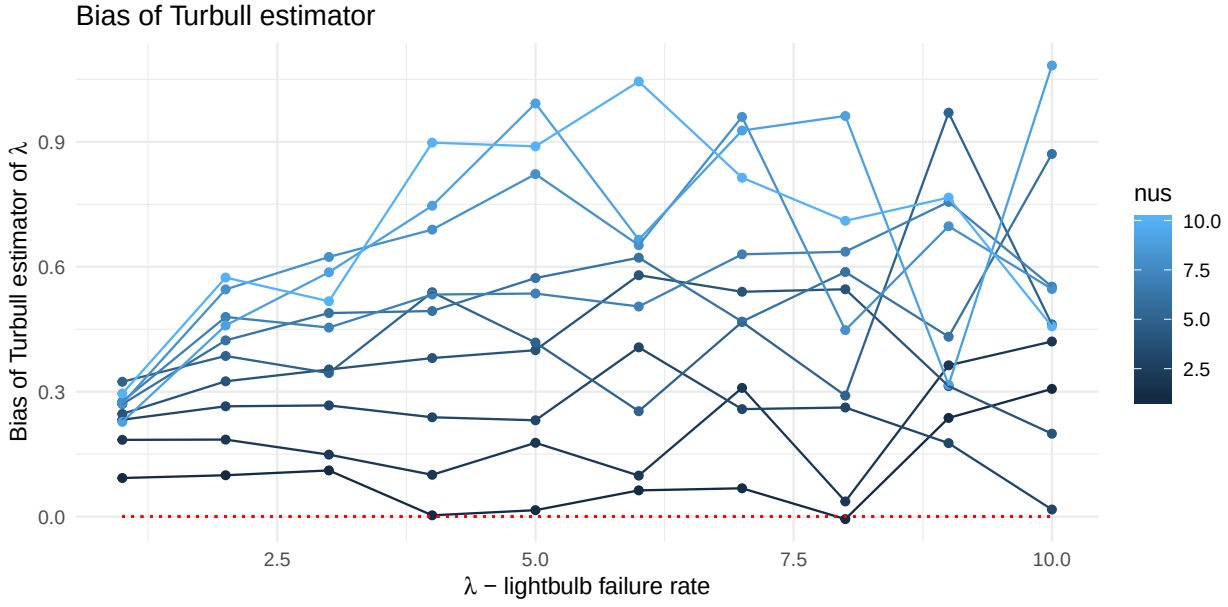

Figure 13: Bias of Turnbull estimator of the failure rate.

Mean square error of the estimator

At figure 14 we observe that the mean square error of Turnbull estimator is bigger for bigger ν and λ values. It is the exponential growth. The values of mean square error of the estimator are much smaller than for the naive estimator 10.

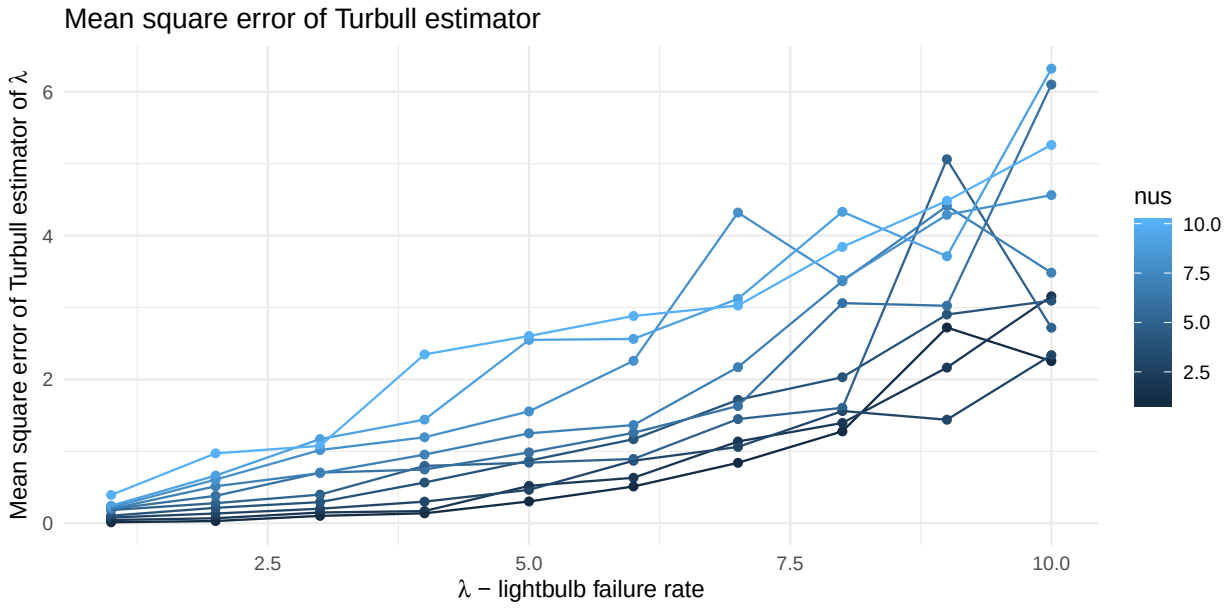

Figure 14: Mean square error of Turnbull estimator of the failure rate.

Conclusions

In this report we created a generator for periodically inspected lightbulb. Based on generated data we estimated failure rate using the naive and Turnbull estimators. For both types of estimators we tried to answer the questions: how the mean of such estimator depends on true failure rate and inspection rate, how the variance of such estimator depends on true failure rate and inspection rate, how the bias of such estimator depends on true failure rate and inspection rate, how the mean square error of such estimator depends on true failure rate and inspection rate.

It seems that Turnbull estimator, the one which takes into account interval censoring, give us closer to real ones results.