03_proc_Hawkes

Frances Lin

4/22/2021

```
library(tidyverse)
library(ggplot2)
library(patchwork)
```

```
# Simuluate HPP
set.seed(1) # for reproducibility

t_max <- 10
t <- 0

lmbda <- 10

t_vec <- numeric(0) # vector of t # consider change it to t_vec

while(t <= t_max){
  u       <- runif(1)
  t       <-  t - log(u)/lmbda                # t ~ exp(1/lambda)
  if(t < t_max) {
    t_vec <- c(t_vec,t)
  }
}
```

```
# Writing the thinning algorithm using James's way

# Initialize
# Note that mu > 0 and 0 < alpha < beta ??
mu = 0.025
alpha = 0.5
beta = 0.7

# Create lambda(t) function
lmbda_fun <- function(time, obs){
  diff = time - obs
  diff = diff[diff > 0]
  a = sum(alpha * exp(-beta * diff))
  out = mu + a
  return(out)
}

# Apply the lmbda_fun function
lmbda_star <- sapply(X = t_vec, FUN = lmbda_fun, obs = t_vec)
length(lmbda_star)
```

```
## [1] 114
```

```r
# lmbda_star #114
```

```r
# t_vec # 114
```

```r
# t_vec[runif(length(prob_keep)) < min(prob_keep, 1)] #75
```

```r
# lmbda_star[runif(length(prob_keep)) < min(prob_keep, 1)] #75
```

```r
set.seed(1)
lmbda <- median(lmbda_star) #10
prob_keep <- lmbda / lmbda_star
#t_keep <- t_vec[runif(length(prob_keep)) < min(prob_keep, 1)]
check <- runif(length(prob_keep)) < min(prob_keep, 1)
lmdba_keep <- lmbda_star[check]
t_keep <- t_vec[check]
length(lmdba_keep)
```

```
## [1] 87
```

```r
length(t_keep)
```

```
## [1] 87
```

```r
length(t_vec)
```

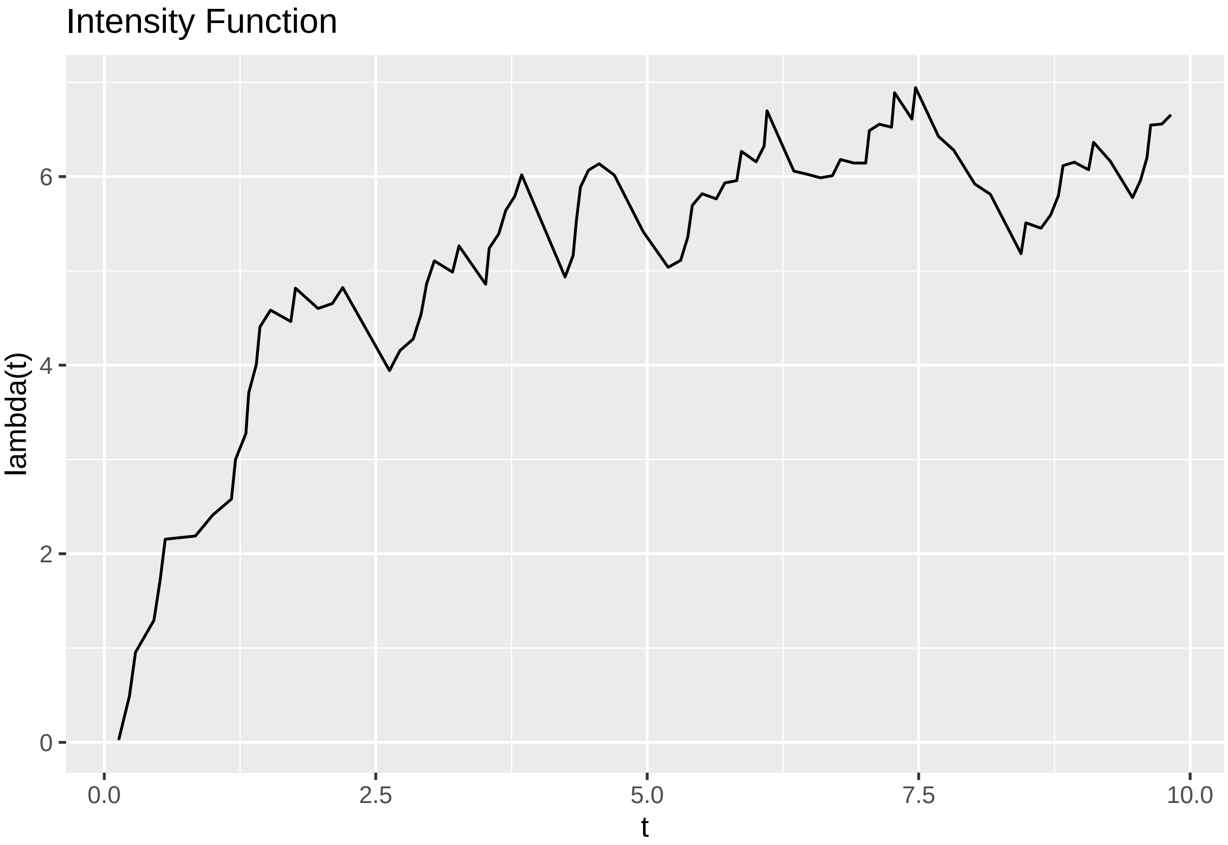
```
## [1] 114
```

```r
#prob_keep
#lmbda_star
```

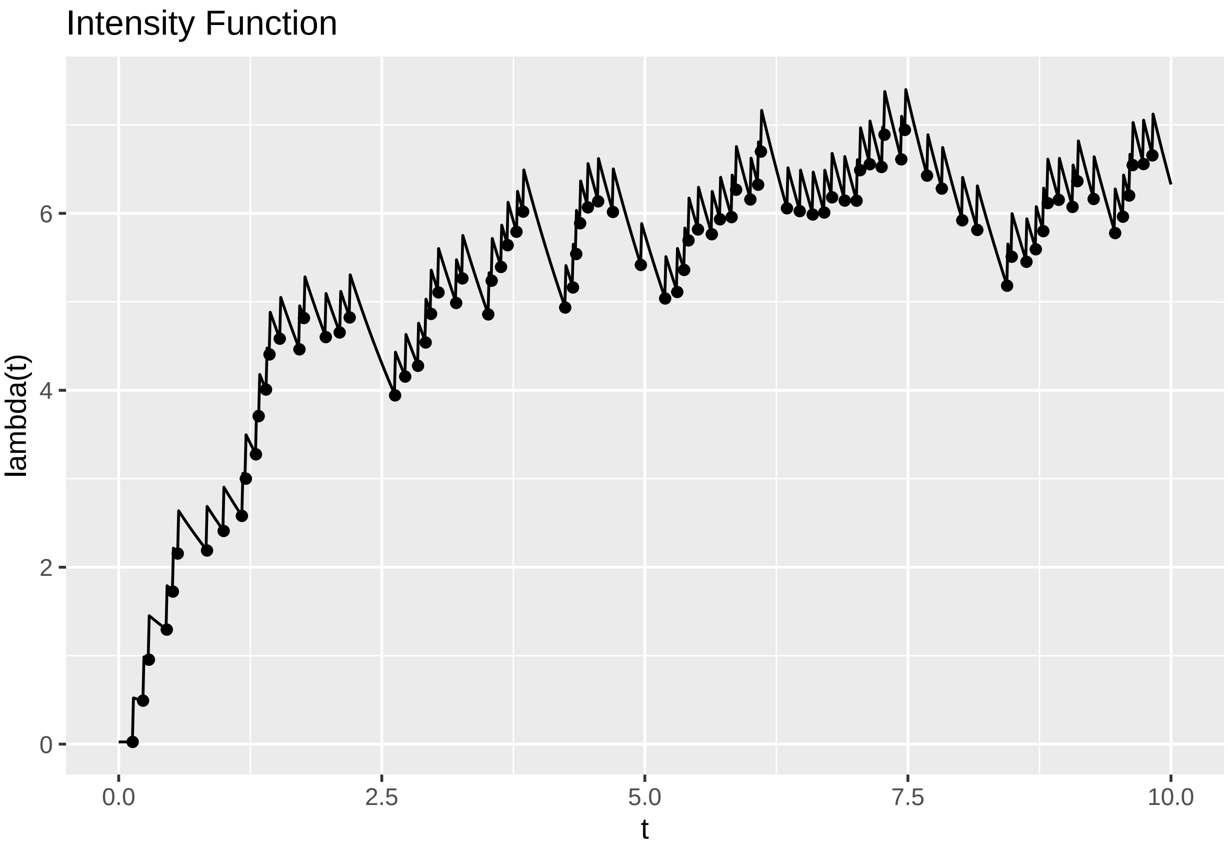
```r
# Plot Hawkes
# Create a df
df_Hawkes = tibble(
  x = t_keep,
  y = 1:(length(t_keep)), #0:(length(X) - 1)
  lmbda = lmdba_keep,
  lmbda2 = sapply(X = t_keep, FUN = lmbda_fun, obs = t_keep)
)

df_Hawkes2 = tibble(
  x = seq(0, 10, length.out = 1001),
  y = sapply(X = x, FUN = lmbda_fun, obs = t_keep)
)


p_Hawkes <- ggplot(data=df_Hawkes, mapping=aes(x=x, y=y)) +
  geom_step() +
  labs(title = "Hawkes Process",
       x = "t",
       y = "N(t)")
#p_Hawkes
```

```
# Plot time plot
p_Hawkes_time <- ggplot(data=df_Hawkes, mapping=aes(x=x, ymin = -0.5, ymax = 0.5)) +
  geom_linerange() +
  geom_hline(aes(yintercept = 0), linetype = "dashed") +
  labs(title = "Corresponding Inter-Arrivial Times",
       x = "t",
       y = "") +
  scale_x_continuous(limits = c(0, 10), breaks = seq(0, 10, by = 1)) +
  theme(axis.text.y=element_blank(), axis.ticks.y=element_blank())
#p_Hawkes_time
```

```
# Plot lambda_star vs t
p_Hawkes_Int <- ggplot(df_Hawkes, aes(x=x, y=lmbda2)) + # y is not right
  geom_line() +
  labs(title = "Intensity Function",
       x = "t",
       y = "lambda(t)")
p_Hawkes_Int
```

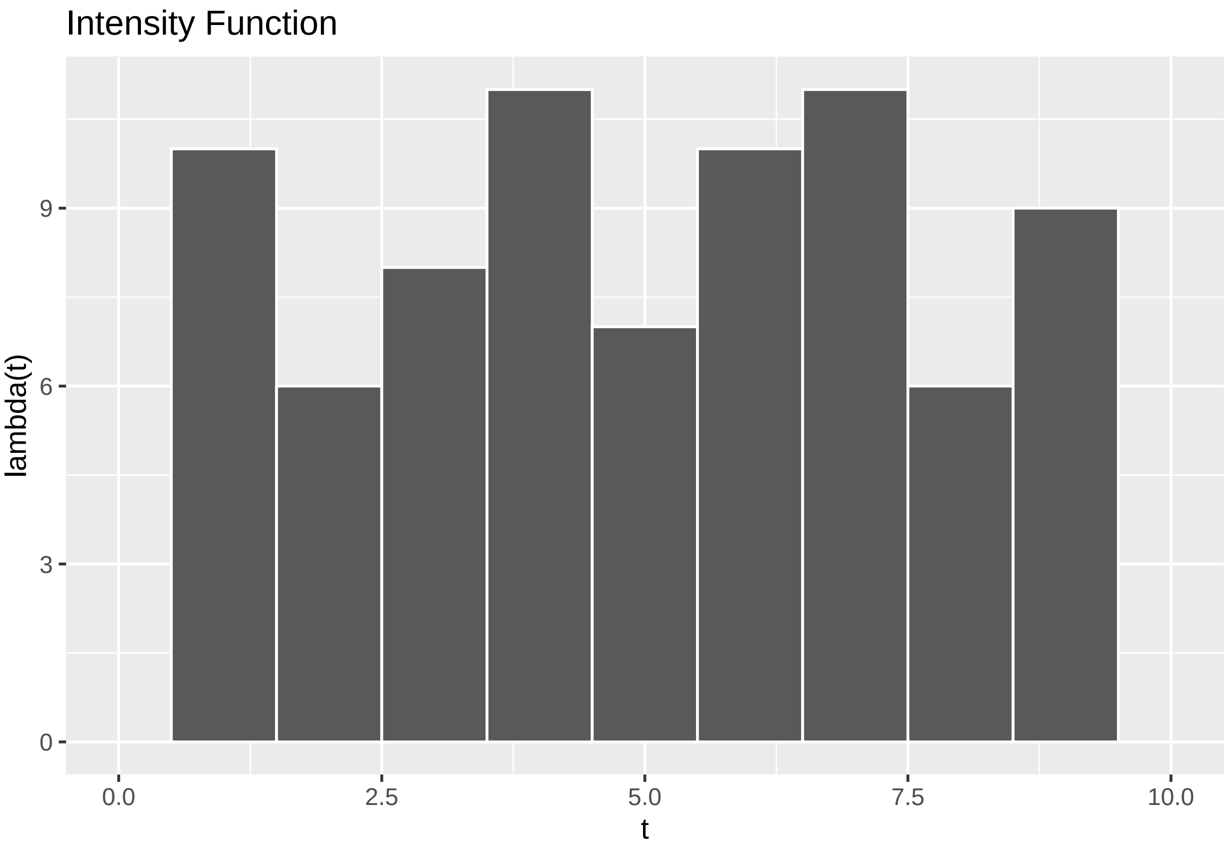


```
p_Hawkes_Int2 <- ggplot(df_Hawkes2, aes(x=x, y=y)) + # y is not right
  geom_line() +
  geom_point(aes(x=x, y=lmbda2), data = df_Hawkes) +
  labs(title = "Intensity Function",
       x = "t",
       y = "lambda(t)")
p_Hawkes_Int2
```

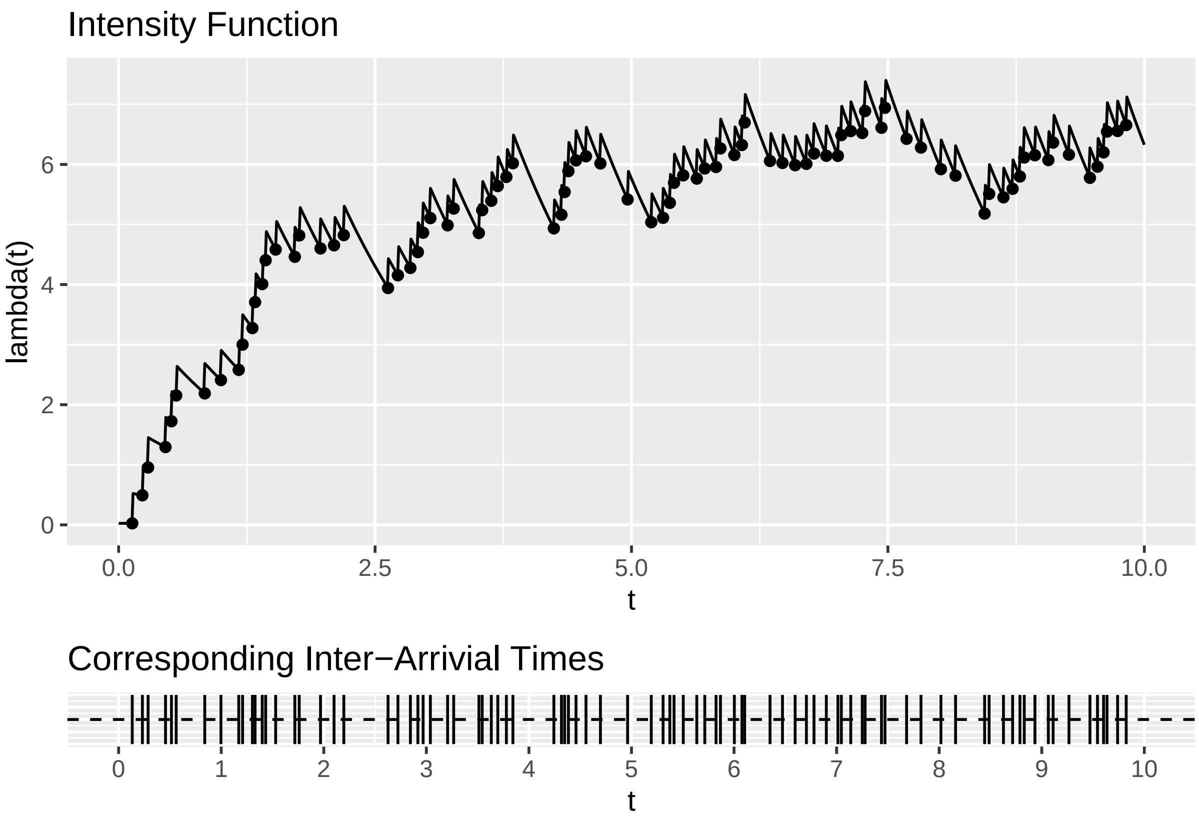


```
# Plot lambda (rate)
p_Hawkes_hist <- ggplot(data=df_Hawkes, mapping=aes(x=x)) +
  geom_histogram(bins = 11, color = "white") +
  labs(title = "Intensity Function",
       x = "t",
       y = "lambda(t)") +
  xlim(0, 10) # so that scale lines up
p_Hawkes_hist
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



```
# # Combine plots
# require(gridExtra)
# grid.arrange(p_Hawkes, p_Hawkes_time)
p_Hawkes_Int2 / p_Hawkes_time + plot_layout(heights = c(0.9, 0.1))
```



Intensity Function

Corresponding Inter−Arrivial Times

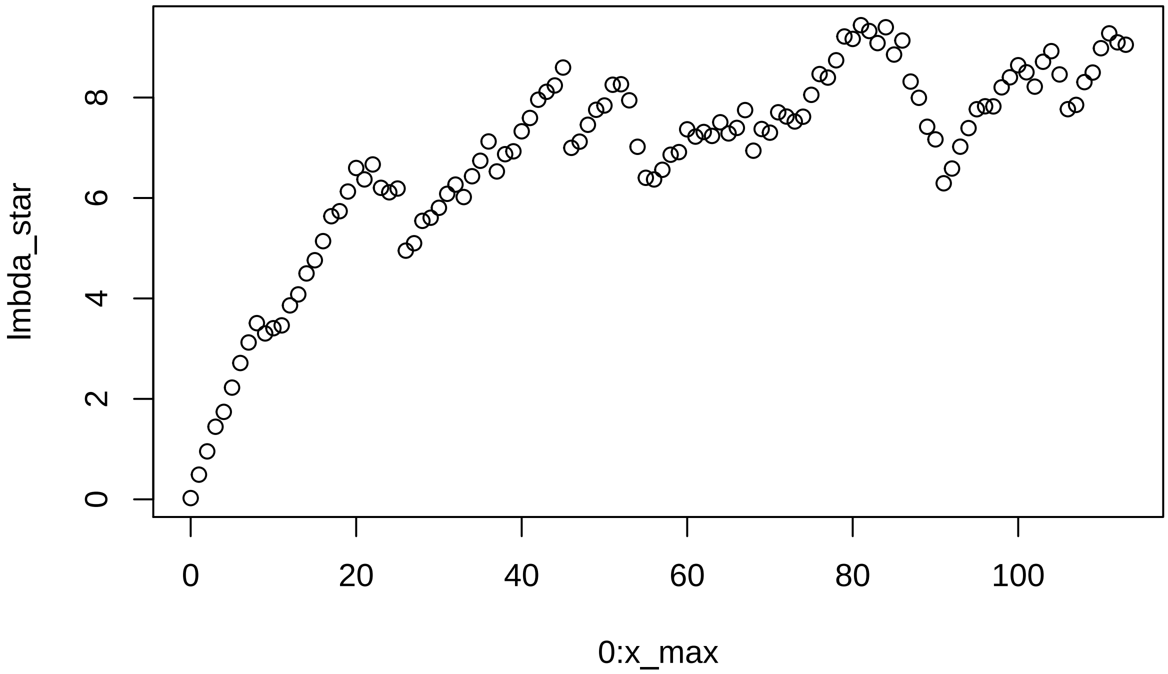
```
# Save output and adjust size
png(file = '/Users/franceslinyc/Hawkes-Process-2021/results/plot_1D_Hawkes.png', width = 450*2, height =
p_Hawkes_Int2 / p_Hawkes_time + plot_layout(heights = c(0.9, 0.1))
```

```
# while(t <= t_max){
#   if(runif(1) < min(lmbda/lmbda_star, 1)){
#     X_keep <- c(X, t)
#   }
#   return(X_keep)
# }
```

```
# Plot lmbda_star
# Not thinned yet
length(lmbda_star) #114
```

```
## [1] 114
```

```
x_max = length(lmbda_star) - 1
plot(x = 0:x_max, y = lmbda_star)
```



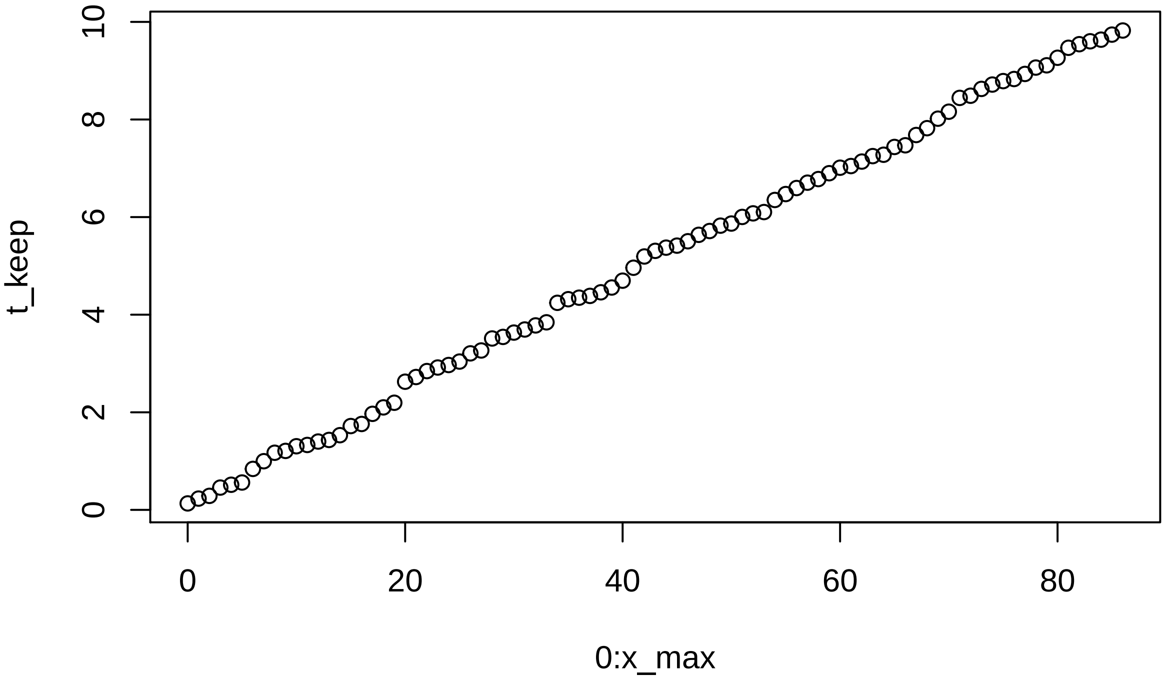
```r
lmbda_star
```

```
##   [1] 0.0250000 0.4915714 0.9546019 1.4449991 1.7414364 2.2248740 2.7141392
##   [8] 3.1229776 3.5081286 3.3028326 3.4072808 3.4634986 3.8613540 4.0804227
##  [15] 4.4977721 4.7607225 5.1405127 5.6373191 5.7370722 6.1285622 6.5974229
##  [22] 6.3700079 6.6678748 6.2021969 6.1130062 6.1884531 4.9519113 5.0987377
##  [29] 5.5445297 5.6070956 5.8042514 6.0833792 6.2670777 6.0187028 6.4331330
##  [36] 6.7410864 7.1256539 6.5289133 6.8721576 6.9291239 7.3275696 7.5933949
##  [43] 7.9563635 8.1139206 8.2402694 8.5971533 6.9987436 7.1215806 7.4577060
##  [50] 7.7564492 7.8414950 8.2549642 8.2649085 7.9449665 7.0195651 6.4015197
##  [57] 6.3691423 6.5617115 6.8614405 6.9138118 7.3668149 7.2221385 7.3138756
##  [64] 7.2359179 7.5075685 7.2853576 7.3951503 7.7498733 6.9420035 7.3731872
##  [71] 7.3009528 7.7062678 7.6215417 7.5229289 7.6183331 8.0539677 8.4673699
##  [78] 8.3969444 8.7418752 9.2159893 9.1669439 9.4408638 9.3248153 9.0840499
##  [85] 9.3996473 8.8558030 9.1357698 8.3182832 7.9949068 7.4179334 7.1667764
##  [92] 6.2928722 6.5863507 7.0213642 7.3913944 7.7676491 7.8259221 7.8237939
##  [99] 8.2028958 8.4028766 8.6435296 8.5028558 8.2171400 8.7126734 8.9237189
## [106] 8.4599969 7.7682505 7.8532729 8.3073663 8.4976505 8.9825152 9.2780350
## [113] 9.0990944 9.0520222
```

```r
# Plot lmbda_star
# Not thinned yet
length(t_keep) # 75
```

```
## [1] 87
```

```r
x_max = length(t_keep) - 1
plot(x = 0:x_max, y = t_keep)
```


t_keep
0
2
4
6
8
10
0
20
40
60
80
0:x_max