Hawkes

Tomasz Dróżdż

```
library(evently)
library(ggplot2)
library(scales)
```

Badam problem rozprzestrzeniania się tweetów, który nadaje się do modelowania procesem Hawkesa z racji tego, iż jest to proces samopobudzające się (popularność tweeta rośnie wraz z jego udostępnianiem), którego intensywność maleje wraz z czasem (bez udostępnień popularność spada).

Jako że mamy dostęp do liczby obserwujących osoby, która udostępnia tweeta, jest to proces znakowany (marked). Wybieram kernel z funkcją power-law (testowałem również funkcję eksponencjalną), gdyż z badań Rizoiu wynika, że działa ona lepiej dla problemów mediów społecznościowych.

```
tweets <- read.csv(file = 'index.csv',
                   colClasses = c('character', 'numeric', 'integer', 'integer'))
retweets <- read.csv(file = 'data.csv', colClasses = c('integer', 'integer'))

tweets$cascade_length = tweets$end_ind - tweets$start_ind + 1
```

Z opisu zbioru:

> We only kept tweets such that it has at least 50 retweets, the text of the tweet does not contain a pound sign # (hashtag), and the language of the original poster is English.

```
cat("Number of tweets:", nrow(tweets))
```

```
## Number of tweets: 166076
```

```
tweets[1:10,]
```

```
##              tweet_id post_time_day start_ind end_ind cascade_length
## 1  122434619336429568     0.9266435         1     175            175
## 2  122449665319911424     0.9681597       176     369            194
## 3  122450173157847040     0.9695602       370     703            334
## 4  122442987455254528     0.9497338       704     827            124
## 5  122456625469591552     0.9873727       828     941            114
## 6  122310349528645632     0.5837269       942    1029             88
## 7  122363390093049856     0.7300926      1030    2418           1389
## 8  122295421430284288     0.5425347      2419    2490             72
## 9  122352576355237888     0.7002546      2491    2545             55
## 10 122434507684052992     0.9263426      2546    2612             67
```

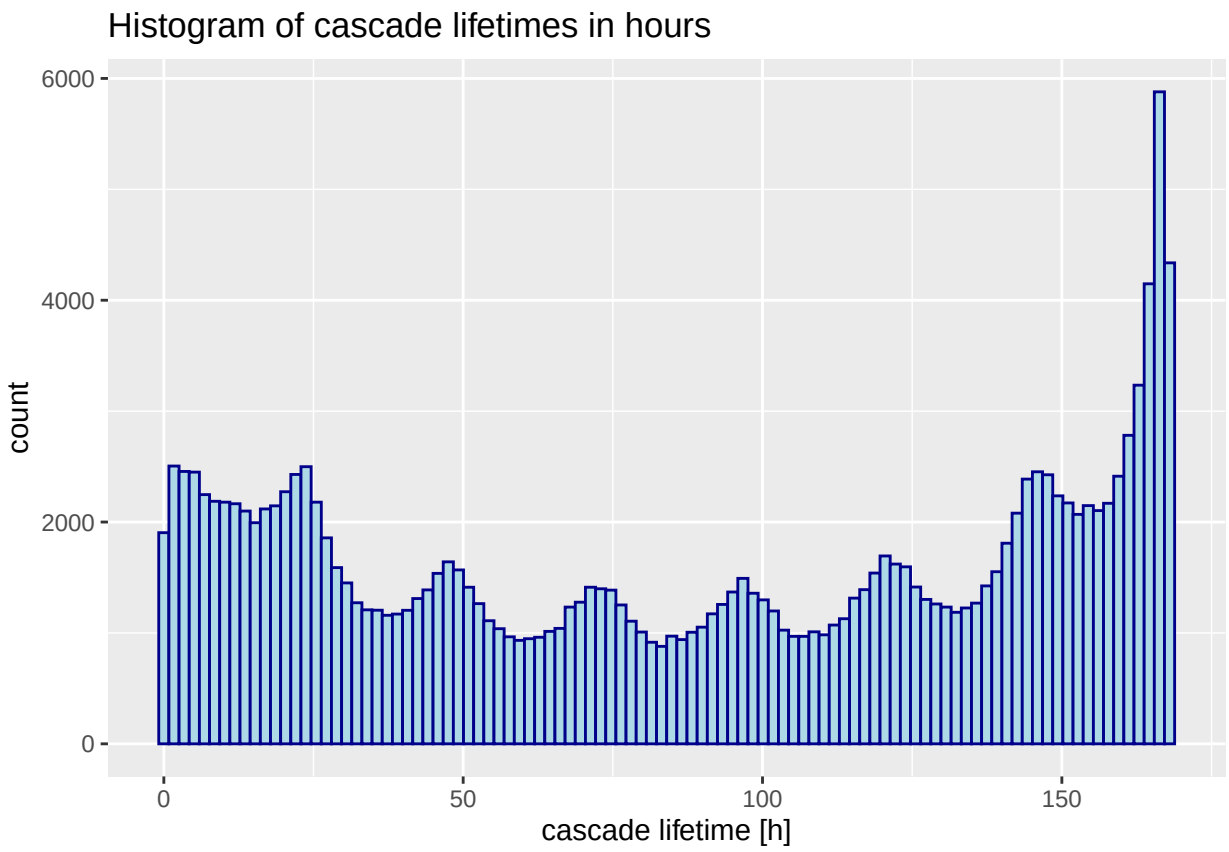
```
cat("Number of retweets:", comma(nrow(retweets)))
```

```
## Number of retweets: 34,784,488
```

```
retweets[1:10,]
```

```
##    relative_time_second number_of_followers
## 1                     0                  33
## 2                 84833               46828
## 3                 84878                 208
## 4                 84883                  37
## 5                 84900                 137
## 6                 84904                 254
## 7                 84905                  95
## 8                 84910                 178
## 9                 84914                  59
## 10                84920                   5
```

```
cascade_lifetime <- retweets[tweets$end_ind, 1] / (60 * 60)
ggplot(as.data.frame(cascade_lifetime), aes(x = cascade_lifetime)) +
  geom_histogram(bins = 100, color = "darkblue", fill = "lightblue") +
  labs(title = "Histogram of cascade lifetimes in hours",
       x = "cascade lifetime [h]")
```



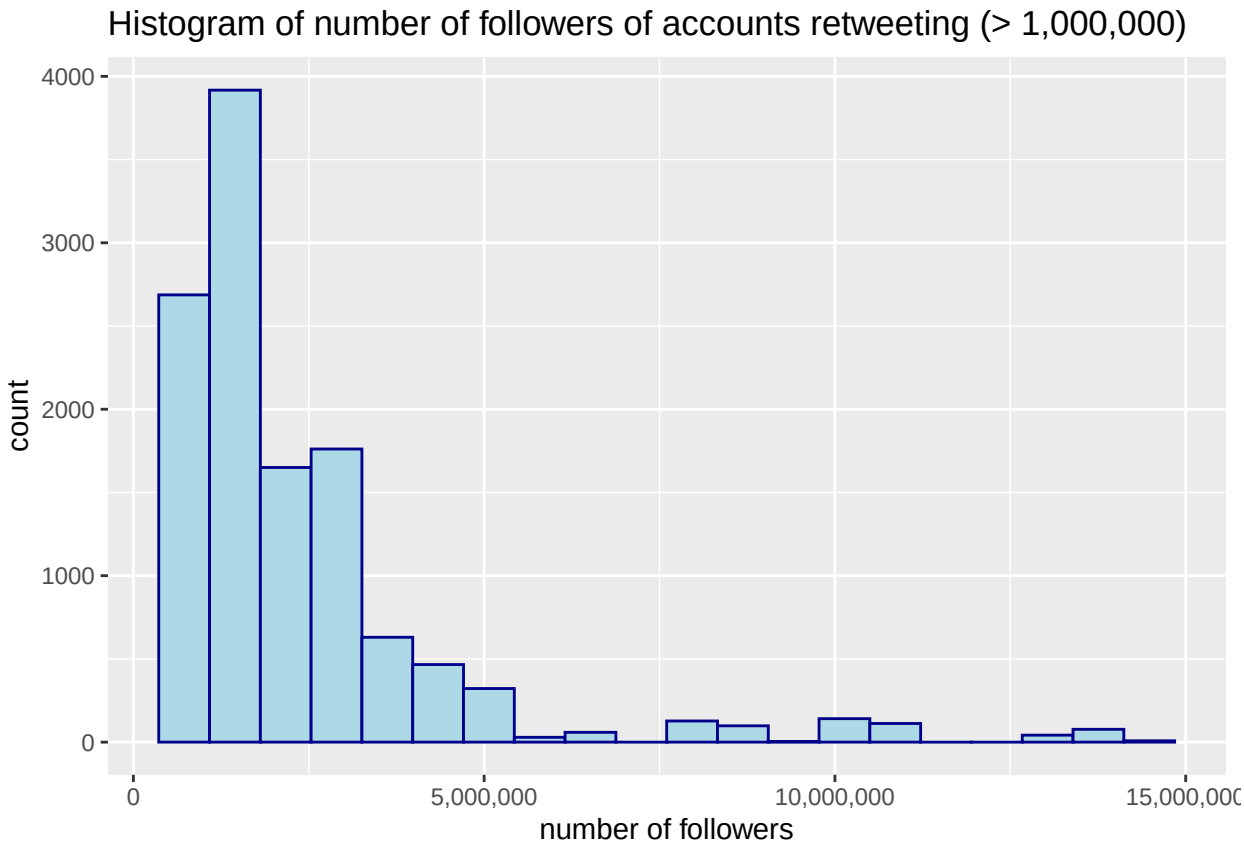
```
cat("Median followers count:", median(retweets$number_of_followers), "\n")
```

```
## Median followers count: 120
```

```
cat("Number of accounts with less than a million followers:",
    comma(nrow(retweets[retweets$number_of_followers < 1000000, ])),
    "\n")
```

```
## Number of accounts with less than a million followers: 34,772,359
```

```
followers <- retweets[retweets$number_of_followers >= 1000000, 2]
ggplot(as.data.frame(followers), aes(x = followers)) +
  geom_histogram(bins = 20, color = "darkblue", fill = "lightblue") +
  labs(title = "Histogram of number of followers of accounts retweeting (> 1,000,000)",
       x = "number of followers") +
  scale_x_continuous(labels = comma)
```




```
prepare_cascade <- function(tweet_idx) {
  tweet <- tweets[tweet_idx, ]
  cascade <- retweets[tweet$start_ind:tweet$end_ind, ]
  colnames(cascade) <- c("time", "magnitude")
  return (cascade[, c(2, 1)])
}
```

```r
prepare_cascade(3)[1:10, ]
```

```
##     magnitude  time
## 370     40627     0
## 371       632    23
## 372       270    38
## 373        67 40507
## 374    434245 66508
## 375        60 66524
## 376       420 66530
## 377        80 66530
## 378        99 66543
## 379        16 66548
```

```r
cat("Median cascade length:", median(tweets$cascade_length))
```

```
## Median cascade length: 111
```

```r
br = seq(0, 34000, by = 1000)

ranges <- paste(head(br, -1), br[-1], sep=" - ")
freq <-  hist(tweets$cascade_length, breaks=br, include.lowest=TRUE, plot=FALSE)

freq_df <- data.frame(cascade_length = ranges, frequency = freq$counts)
freq_df[freq_df$frequency > 0, ]
```

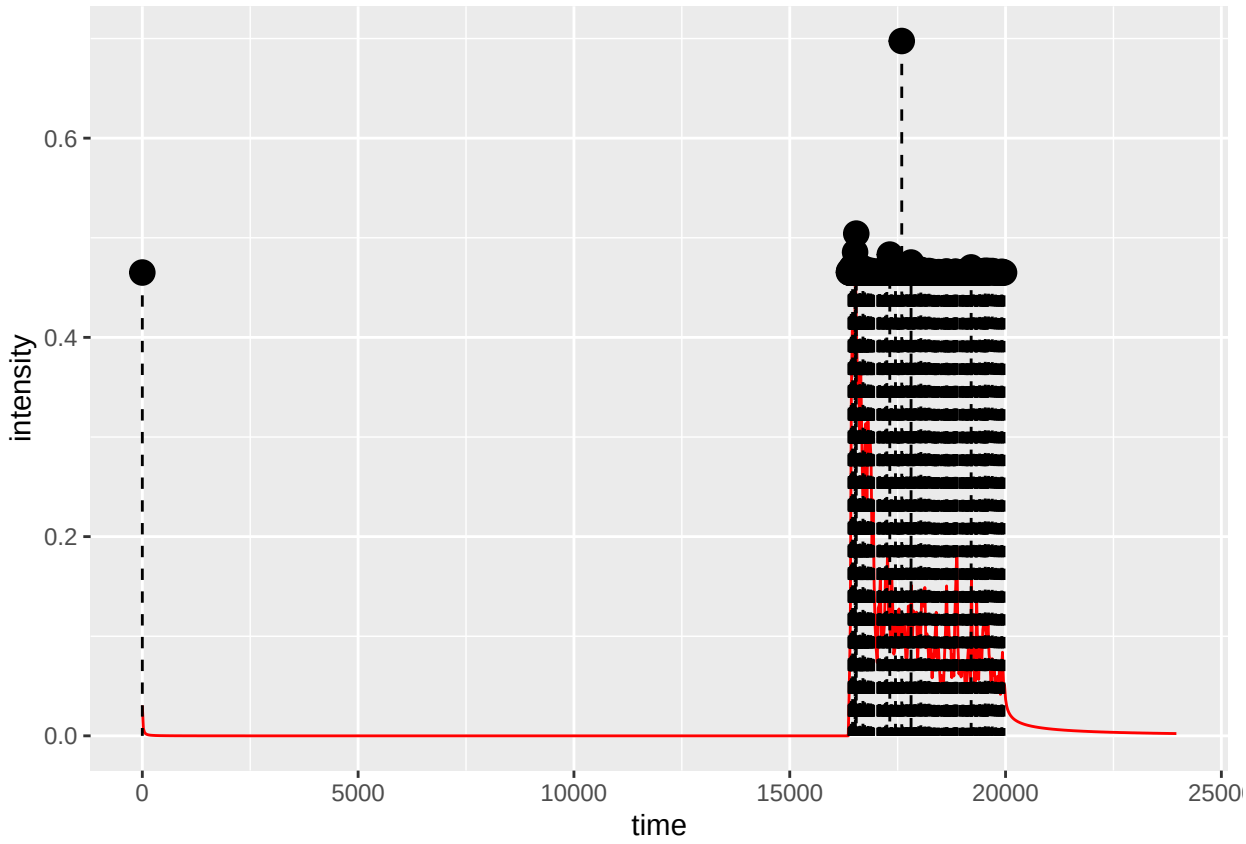
```
##    cascade_length frequency
## 1          0 - 1000    162940
## 2       1000 - 2000      2398
## 3       2000 - 3000       396
## 4       3000 - 4000       127
## 5       4000 - 5000        64
## 6       5000 - 6000        39
## 7       6000 - 7000        18
## 8       7000 - 8000        21
## 9       8000 - 9000        18
## 10     9000 - 10000        12
## 11    10000 - 11000         9
## 12    11000 - 12000         7
## 13    12000 - 13000         8
## 14    13000 - 14000         2
## 15    14000 - 15000         2
## 16    15000 - 16000         5
## 17    16000 - 17000         5
## 18    17000 - 18000         2
## 20    19000 - 20000         1
## 22    21000 - 22000         1
## 34    33000 - 34000         1
```

```r
fit_model <- function(cascade) {
  train_time <- cascade[2, 2] + (60 * 60)
  train_rows <- cascade[cascade$time < train_time,]

  cat("Fitting cascade of length", nrow(cascade), "\n")
  cat("Using first",
      comma(train_time),
      "seconds for training -",
      nrow(train_rows),
      "tweets\n\n")

  fitted_model <-
    fit_series(
      train_rows,
      model_type = 'mPL',
      observation_time = max(train_rows$time),
      cores = 20
    )

  branching_factor <- get_branching_factor(fitted_model)
  cat("Branching factor:", branching_factor, "\n\n")

  if (branching_factor < 1) {
    predicted_popularity <- predict_final_popularity(fitted_model)
    real_popularity <- nrow(cascade)

    cat("Predicted final popularity:", predicted_popularity, "\n")
    cat("Real final popularity:", real_popularity, "\n")
    cat("Relative error:",
        sprintf(
          "%0.2f%%",
          100 * abs(predicted_popularity - real_popularity) / real_popularity
        ),
        "\n\n")
  }

  return (fitted_model)
}
```

```r
cascade_1 <- prepare_cascade(28)
model_1 <- fit_model(cascade_1)
```

```
## Fitting cascade of length 872
## Using first 19,971 seconds for training - 456 tweets
##
## Branching factor: 0.8035607
##
## Predicted final popularity: 740.2078
## Real final popularity: 872
## Relative error: 15.11%
```

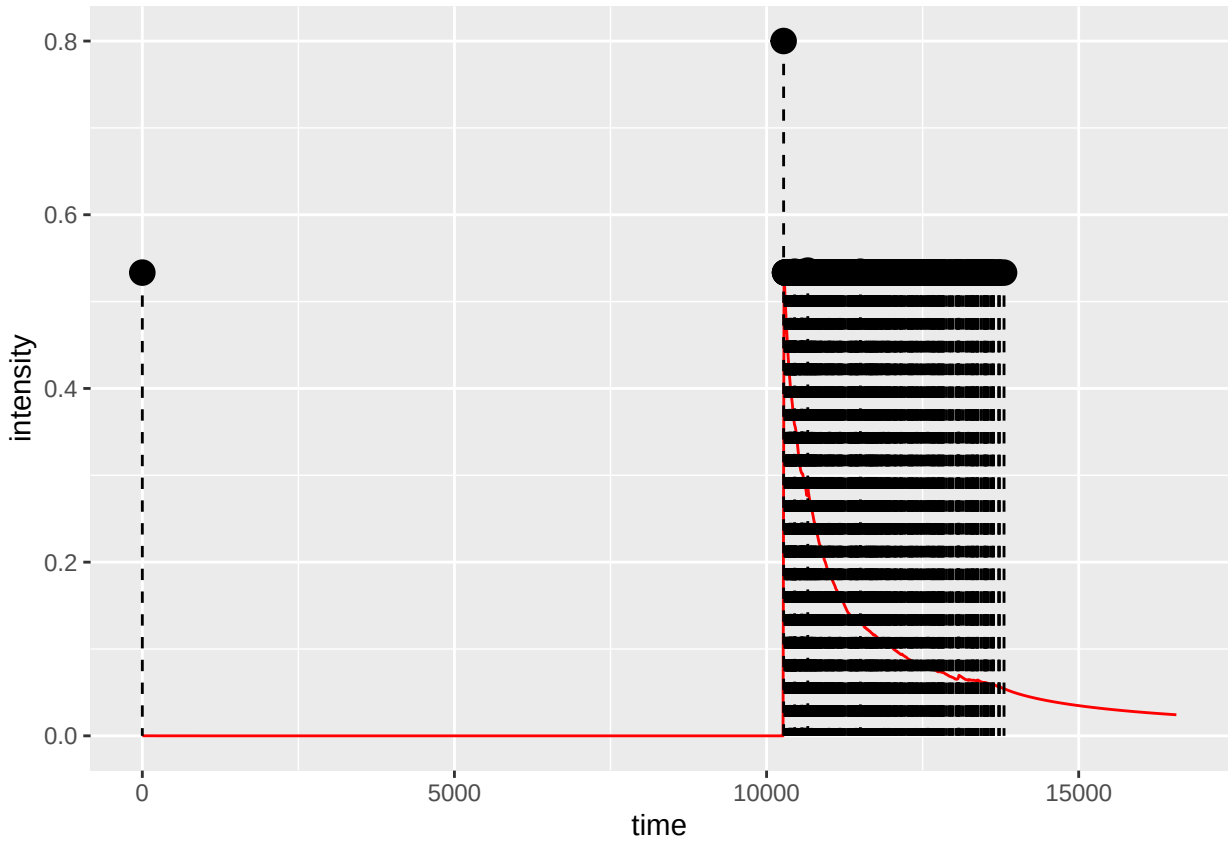
```r
plot_event_series(model_1)
```



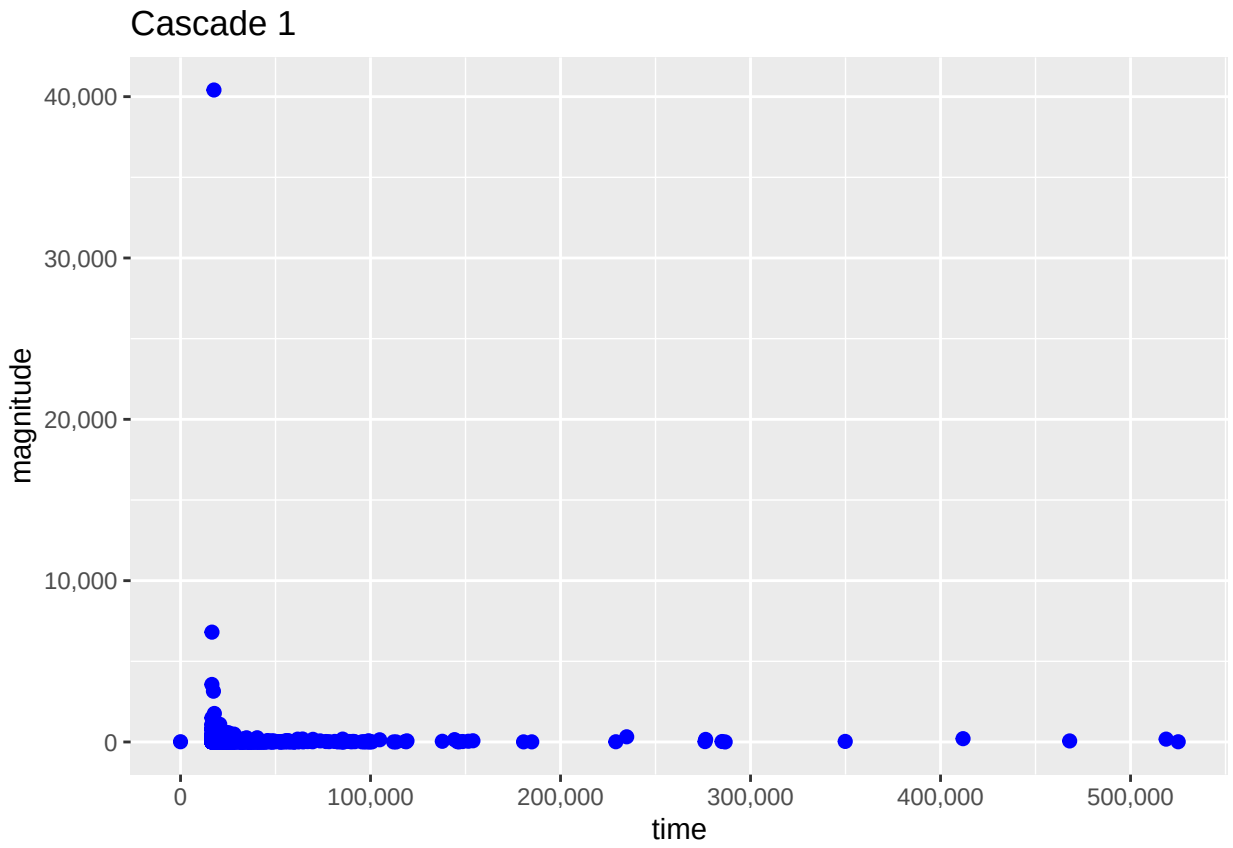

```
cascade_2 <- prepare_cascade(45)
model_2 <- fit_model(cascade_2)
```

```
## Fitting cascade of length 792
## Using first 13,873 seconds for training - 487 tweets
##
## Branching factor: 0.0850471
##
## Predicted final popularity: 1617.731
## Real final popularity: 792
## Relative error: 104.26%
```

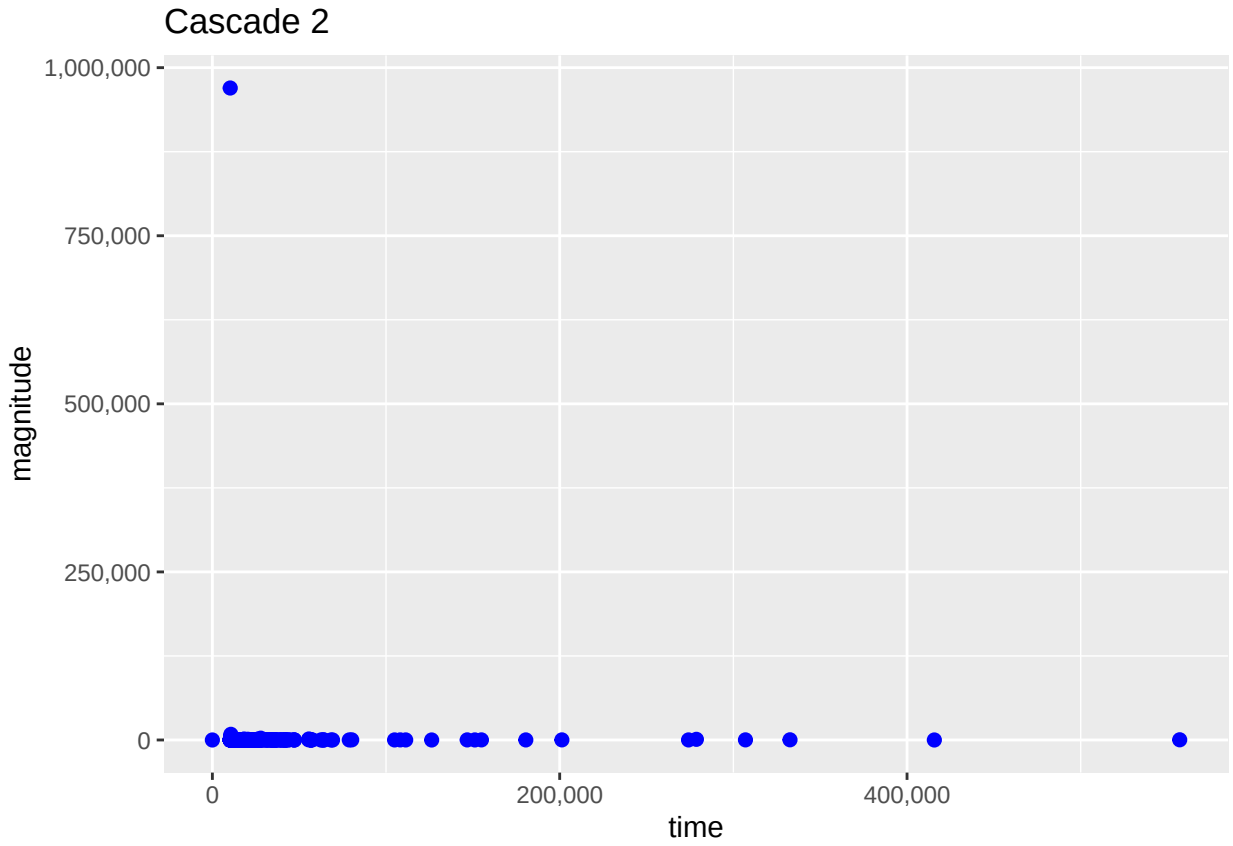
```
plot_event_series(model_2)
```




```
ggplot(cascade_1, aes(x=time, y=magnitude)) + geom_point(size=2, color="blue") +
  scale_x_continuous(labels = comma) + scale_y_continuous(labels = comma) +
  labs(title="Cascade 1")
```




```
ggplot(cascade_2, aes(x=time, y=magnitude)) + geom_point(size=2, color="blue") +
  scale_x_continuous(labels = comma) + scale_y_continuous(labels = comma) +
  labs(title="Cascade 2")
```




```
comma(model_1$par)
```

```
##       K    beta       c   theta
##  "1.21"  "0.11" "15.19"  "0.47"
```

```
comma(model_2$par)
```

```
##        K     beta        c    theta
##   "0.01"   "0.79" "300.00"   "0.16"
```

Wartość K modelu trenowanego na pierwszej kaskadzie jest znacznie większa, niż wartość K modelu trenowanego na drugiej kasadzie, co oznacza, że dany tweet będzie szybciej się rozprzestrzeniał (zwiększa intensywność procesu o większą wartość, jako że jest ona skalowana przez K).

Wartość *beta* jest większa dla modelu wytrenowanego na drugiej kaskadzie, co oznacza, że liczba obserwujących osobę retweetującą ma w nim większe znaczenie i będzie w większym stopniu zwiększać intensywność procesu.

Wartość parametru c jest większa dla modelu wytrenowanego na drugiej kasadzie, co sugeruje, że znajduje się w niej więcej zbliżonych w czasie zdarzeń przy wysokiej wartości funkcji intensywności (c wprowadza przesunięcie, aby ogarniczyć wartość funkcji intensywności, gdy zdarzenia następują w krótkich odstępach czasu).

Model wytrenowany na drugiej kaskadzie ma większą wartość *thety*, co oznacza, że zdarzenia są szybciej zapominane (intensywność procesu szybciej spada wraz z upływem czasu).