

# Bayesian estimation for NHPP using **rstan**

*Miao Cai miao.cai@slu.edu*

*2019-07-08*

## Contents

<b>1</b>	<b>Reproduce the error</b>	<b>3</b>
1.1	An example that works . . . . .	4
1.2	An example that does not work . . . . .	6
<b>2</b>	<b>Update code</b>	<b>7</b>

The Stan code I used previously cannot estimate trips without any critical event. In this document, I will try to solve this issue.

```
source("functions/NHPP_functions.R")
require(rstan)
```

```
## Loading required package: rstan
```

```
## Loading required package: ggplot2
```

```
## Registered S3 methods overwritten by 'ggplot2':
```

```
##   method      from
##   [.quosures   rlang
##   c.quosures   rlang
##   print.quosures rlang
```

```
## Loading required package: StanHeaders
```

```
## rstan (Version 2.18.2, GitRev: 2e1f913d3ca3)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
```

```
## options(mc.cores = parallel::detectCores()).
```

```
## To avoid recompilation of unchanged Stan programs, we recommend calling
```

```
## rstan_options(auto_write = TRUE)
```

```
rstan_options(auto_write = TRUE)
```

# 1 Reproduce the error

```
plptau = '  
functions{  
  real nhpp_log(vector t, real beta, real theta, real tau){  
    vector[num_elements(t)] loglik_part;  
    real loglikelihood;  
    for (i in 1:num_elements(t)){  
      loglik_part[i] = log(beta) - beta*log(theta) + (beta - 1)*log(t[i]);  
    }  
    loglikelihood = sum(loglik_part) - (tau/theta)^beta;  
    return loglikelihood;  
  }  
}  
data {  
  int<lower=0> N; //total # of obs  
  int<lower=0> K; //total # of groups  
  vector<lower=0>[K] tau;//truncated time  
  vector<lower=0>[N] event_time; //failure time  
  int s[K]; //group sizes  
}  
parameters{  
  real<lower=0> beta;  
  real<lower=0> theta;  
}  
model{  
  int position;  
  position = 1;  
  for (k in 1:K){  
    segment(event_time, position, s[k]) ~ nhpp(beta, theta, tau[k]);  
    position = position + s[k];  
  }  
}  
//PRIORS  
beta ~ gamma(1, 1);  
theta ~ gamma(1, 0.01);  
}  
,
```

## 1.1 An example that works

Firstly, I create a example with sufficiently long time intervals so that there are at least 1 event in each time interval.

```
df0 = list(  
  N = 38L, K = 10L,  
  tau = c(21.269, 18.109, 19.468, 19.89, 18.247, 19.048, 19.957, 21.006,  
17.524, 19.475),  
  event_time = c(5.045, 14.921, 18.566, 7.265, 10.51, 12.155, 16.262, 17.738,  
17.763, 16.059, 18.371, 10.393, 11.787, 5.088, 10.144, 11.646,  
13.274, 15.233, 16.345, 17.583, 15.266, 15.391, 16.355, 17.79,  
7.729, 13.906, 14.287, 12.012, 18.662, 5.654, 5.727, 8.144, 11.608,  
14.756, 14.933, 16.088, 16.45, 18.876),  
  s = c(3L, 6L, 2L, 2L, 7L, 4L, 3L, 2L, 6L, 3L)  
)  
df0
```

```
## $N  
## [1] 38  
##  
## $K  
## [1] 10  
##  
## $tau  
## [1] 21.269 18.109 19.468 19.890 18.247 19.048 19.957 21.006 17.524 19.475  
##  
## $event_time  
## [1] 5.045 14.921 18.566 7.265 10.510 12.155 16.262 17.738 17.763 16.059  
## [11] 18.371 10.393 11.787 5.088 10.144 11.646 13.274 15.233 16.345 17.583  
## [21] 15.266 15.391 16.355 17.790 7.729 13.906 14.287 12.012 18.662 5.654  
## [31] 5.727 8.144 11.608 14.756 14.933 16.088 16.450 18.876  
##  
## $s  
## [1] 3 6 2 2 7 4 3 2 6 3
```

Stan works out well and produce estimates close to the true parameters  $\beta = 2, \theta = 10$ .

```
fitplp <- stan(  
  model_code=plptau, model_name="NHPP", data=df0,  
  iter=1000, warmup = 500, chains=1, seed = 123, refresh = 0
```

```
)
fitplp
```

```
## Inference for Stan model: NHPP.
## 1 chains, each with iter=1000; warmup=500; thin=1;
## post-warmup draws per chain=500, total post-warmup draws=500.
##
##          mean se_mean   sd  2.5%   25%   50%   75%  97.5% n_eff Rhat
## beta      2.16     0.03 0.37   1.53   1.91   2.14   2.37   2.98   115 1.00
## theta    10.53     0.12 1.40   7.72   9.59  10.58  11.51  13.16   130 1.00
## lp__    -90.13     0.09 1.15 -93.08 -90.57 -89.78 -89.30 -88.98   171 1.03
##
## Samples were drawn using NUTS(diag_e) at Mon Jul  8 11:01:37 2019.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

## 1.2 An example that does not work

However, if there is no event in any of the time intervals, the **Stan** code will not work. For example:

```
df1 = list(
  N = 8L, K = 10L,
  tau = c(9.9785, 7.3146, 10.0518, 10.2853, 10.2621, 8.4175, 10.7142, 12.0679, 10.6844, 8.2966),
  event_time = c(6.7346, 8.1608, 4.4621, 6.5118, 7.9538, 11.2332, 11.6206, 11.9121),
  s = c(0L, 0L, 2L, 1L, 2L, 0L, 0L, 3L, 0L, 0L)
)
df1

## $N
## [1] 8
##
## $K
## [1] 10
##
## $tau
## [1] 9.9785 7.3146 10.0518 10.2853 10.2621 8.4175 10.7142 12.0679
## [9] 10.6844 8.2966
##
## $event_time
## [1] 6.7346 8.1608 4.4621 6.5118 7.9538 11.2332 11.6206 11.9121
##
## $s
## [1] 0 0 2 1 2 0 0 3 0 0

fitplp <- stan(
  model_code=plptau, model_name="NHPP", data=df1,
  iter=1000, warmup = 500, chains=1, seed = 123, refresh = 0
)

## Error in sampler$call_sampler(args_list[[i]]) : Initialization failed.
## character(0)

## error occurred during calling the sampler; sampling not done
```

## 2 Update code