Bayesian hierarchical models for NHPP using `rstan`

Miao Cai miao.cai@slu.edu

2019-07-10

Contents

1 Model setting

Let $T_{d,s,i}$ denote the time to the d-th driver's s-th shift's i-th critical event. The total number critical events of d-th driver's s-th shift is $n_{d,s}$. The ranges of these notations are:

- $i = 1, 2, \cdots, n_{d,S_d}$,
- $s = 1, 2, \cdots, S_d$,
- $d = 1, 2, \cdots, D$.

We assume the times of critical events within the d-th driver's s-th shift were generated from a non-homogeneous Poisson process (NHPP) with a power law process (PLP), with a fix rate parameter β and varying scale parameters $\theta_{d,s}$ across drivers. The data generating process is then:

$$
\begin{aligned}
T_{d,s,1}, T_{d,s,2}, \cdots, T_{d,s,n_{d,s}} &\sim \text{PLP}(\beta, \theta_{d,s}) \\
\beta &\sim \text{Gamma}(1, 1) \\
\log \theta_{d,s} &= \gamma_{0d} + \gamma_1 x_{d,s,1} + \gamma_2 x_{d,s,2} + \cdots + \gamma_k x_{d,s,k} \\
\gamma_{01}, \gamma_{02}, \cdots, \gamma_{0D} &\sim \text{i.i.d. } N(\mu_0, \sigma_0^2) \\
\gamma_1, \gamma_2, \cdots, \gamma_k &\sim \text{i.i.d. } N(0, 10^2) \\
\mu_0 &\sim N(0, 10^2) \\
\sigma_0 &\sim \text{Gamma}(1, 1)
\end{aligned}
$$

2 Simulating data

2.1 Theoretical data generating process (DGP)

1. Random intercepts $\gamma_{01}, \gamma_{02}, \cdots, \gamma_{0D}$. The standard deviation of μ_0 was intentionally set to small number 2 to make $\theta_{d,s}$ fall into a reasonably small range. If I otherwise set it as 10, $\theta_{d,s}$ may be more than 10^5 due to the exponentiation, which may not be realistic in real-life data.

$$\mu_0 = 0, \quad \sigma_0 = 0.5$$
$$\sigma_0 \sim \text{Gamma}(1, 1)$$
$$\gamma_{01}, \gamma_{02}, \cdots, \gamma_{0D} \sim \text{i.i.d. } N(\mu_0, \sigma_0^2)$$

2. Fixed parameters: 3 fixed parameters $\gamma_1, \gamma_2, \gamma_3$.

$$\gamma_1, \gamma_2, \gamma_3 \sim \text{i.i.d. } N(0, 0.5^2)$$

3. The number of observations in the d-th driver: N_d.

$$N_d \sim \text{Poisson}(10)$$

4. Data: 3 predictor variables $x_{d,s,1}, x_{d,s,2}, x_{d,s,3}$.

$$x_{d,s,1} \sim \text{N}(0, 1)$$
$$x_{d,s,2} \sim \text{Gamma}(1, 1)$$
$$x_{d,s,3} \sim \text{Poisson}(0.2)$$

5. Scale parameters of a NHPP (random effects): $\theta_{d,s}$.

$$\theta_{d,s} = \text{EXP}(\gamma_{0d} + \gamma_1 x_{d,s,1} + \gamma_2 x_{d,s,2} + \gamma_k x_{d,s,3})$$

6. Shape parameter of a NHPP (fixed effect): $\beta \sim \text{Gamma}(1, 1)$. Set

$$\beta = 1.5$$

7. Simulate a NHPP based on β and $\theta_{d,s}$.

$$T_{d,s,1}, T_{d,s,2}, \cdots, T_{d,s,n_{d,s}} \sim \text{PLP}(\beta, \theta_{d,s})$$

2.2 R code to simulate data and parameters according to the DGP

```r
pacman::p_load(rstan, tidyverse, data.table)
source("functions/NHPP_functions.R")

set.seed(123)
D = 10 # the number of drivers
K = 3 # the number of predictor variables

# 1. Random-effect intercepts
# hyperparameters
mu0 = 0
sigma0 = 0.5
r_0D = rnorm(D, mean = mu0, sd = sigma0)

# 2. Fixed-effects parameters
R_K = rnorm(K, mean = 0, sd = 0.5)

# 3. The number of observations (shifts) in the $d$-th driver: $N_{d}$
N_K = rpois(D, 10)
N = sum(N_K) # the total number of obs
id = rep(1:D, N_K)

# 4. Generate data: x_1, x_2, .. x_K
sim1 = function(group_sizes = N_K){
  ntot = sum(group_sizes)

  int1 = rep(1, ntot)
  x1 =  rnorm(ntot, 0, 1)
  x2 = rgamma(ntot, 1, 1)
  x3 =  rpois(ntot, 0.2)

  return(data.frame(int1, x1, x2, x3))
}
X = sim1(N_K)

# 5. Scale parameters of a NHPP
# 5a. parameter matrix: P
P = cbind(r0 = rep(r_0D, N_K), t(replicate(N, R_K)))
M_logtheta = P*X

# returned parameter for each observed shift
beta = 1.5
theta = exp(rowSums(M_logtheta))
round(theta, 3)
```

```
##  [1] 0.284 1.721 1.440 0.403 1.672 1.265 0.795 2.269 1.485 1.498 2.208
## [12] 1.502 0.875 0.817 0.713 0.777 1.063 0.626 9.235 4.786 1.793 3.045
## [23] 2.627 3.822 2.729 2.755 2.249 2.171 6.624 2.214 6.020 0.975 3.326
## [34] 1.140 2.079 1.334 1.228 0.852 0.792 0.560 1.887 1.434 1.364 2.426
## [45] 4.139 0.976 0.270 5.343 1.880 2.676 4.558 3.342 1.340 3.739 1.500
## [56] 1.794 1.804 1.680 1.937 1.783 0.728 2.232 0.790 0.474 1.108 1.203
## [67] 0.910 0.646 0.507 1.667 0.597 2.714 1.961 0.772 0.441 0.662 1.144
## [78] 0.740 0.659 0.568 0.916 0.606
```

2.3 Generate NHPP data to pass to rstan

```
sim_hier_plp_tau = function(){
  t_list = list()
  len_list = list()
  tau_vector = rnorm(N, 10, 1.3)

  for (i in 1:N) {
    t_list[[i]] = sim_plp_tau(tau_vector[i], beta, theta[i])
    len_list[[i]] = length(t_list[[i]])
  }

  event_dat = data.frame(
    shift_id = rep(1:N, unlist(len_list)),
    event_time = Reduce(c, t_list)
  )

  start_end_dat = data.frame(
    shift_id = 1:N,
    start_time = rep(0, N),
    end_time = tau_vector #difference2
  )

  return(list(event_dat = event_dat,
              start_end_dat = start_end_dat,
              shift_length = unlist(len_list)))
}

df = sim_hier_plp_tau()

hier_dat = list(
    N = nrow(df$event_dat),
    K = nrow(df$start_end_dat),
    D = id, #driver index
    tau = df$start_end_dat$end_time,
    event_time = df$event_dat$event_time,
    s = df$shift_length, #the number of events in each shift
    x1 = X[,2], x2 = X[,3], x3 = X[,4]
)
```

3 Stan code

```
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 shifts
  int<lower=0> D[K];//driver index, this must be an array
  vector<lower=0>[K] tau;//truncated time
  vector<lower=0>[N] event_time; //failure time
  int s[K]; //group sizes
  vector[K] x1;
  vector[K] x2;
  vector[K] x3;
}
parameters{
  real<lower=0> beta;
  vector[K] r0; // random intercept
  vector[3] r; // fixed parameters
  real mu0; // hyperparameter
  real<lower=0> sigma0;// hyperparameter
}
transformed parameters{
  vector<lower=0>[K] theta;
  for (k0 in 1:K){
    theta[k0] = exp(r0[ D[k0] ] + x1[k0]*r[1] + x2[k0]*r[2] + x3[k0]*r[3]);
  }
}
model{
  int position;
  position = 1;
  for (k in 1:K){
    if(s[k] == 0) continue;
    segment(event_time, position, s[k]) ~ nhpp(beta, theta[k], tau[k]);
    position = position + s[k];
  }
  beta ~ gamma(1, 1);
  r0 ~ normal(mu0, sigma0);
  r  ~ normal(0, 10);
  mu0 ~ normal(0, 10);
  sigma0 ~ gamma(1, 1);
  theta ~ gamma(1, 0.01);
}
```

4 Estimated results

4.1 A single simulation to demonstrate

```
f = stan("stan/nhpp_plp_tau_ML.stan",
         chains = 1, iter = 1000, data = hier_dat, refresh = 0)
```

```
## DIAGNOSTIC(S) FROM PARSER:
## Info (non-fatal):
## Left-hand side of sampling statement (~) may contain a non-linear transform of a parameter or local v
## If it does, you need to include a target += statement with the log absolute determinant of the Jacobi
## Left-hand-side of sampling statement:
##     theta ~ gamma(...)

## Warning: There were 1 chains where the estimated Bayesian Fraction of Missing Information was low. Se
## http://mc-stan.org/misc/warnings.html#bfmi-low

## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
pacman::p_load(magrittr)
est = broom::tidy(f)

pull_est = function(var = "theta", est_obj = f){
  z = est_obj %>%
    broom::tidy() %>%
    filter(grepl(var, term)) %>%
    pull(estimate) %>%
    round(3)
  return(z)
}
```

Estimated values:

- Hyperparameters: $\hat{\mu}_0$: 0.043, $\hat{\sigma}_0$: 0.572
- Individual level parameters: $\gamma_1, \gamma_2, \gamma_3$: 0.626, 0.16, 0.164
- Rate parameter β: 1.499
- θ: 0.292, 1.756, 1.416, 0.417, 1.796, 1.308, 0.816, 2.228, 1.567, 1.572, 2.176, 1.529, 0.899, 0.829, 0.73, 0.768, 1.056, 0.605, 9.656, 4.971, 1.69, 2.911, 2.539, 3.928, 2.675, 2.812, 2.295, 2.221, 6.722, 2.225, 6.253, 0.963, 3.417, 1.15, 1.977, 1.351, 1.167, 0.856, 0.747, 0.555, 1.78, 1.449, 1.323, 2.403, 4.42, 0.991, 0.272, 5.425, 1.862, 2.518, 4.72, 3.214, 1.322, 3.689, 1.433, 1.724, 1.788, 1.555, 1.947, 1.66, 0.734, 2.118, 0.796, 0.475, 1.14, 1.195, 0.911, 0.655, 0.481, 1.632, 0.557, 2.695, 1.911, 0.712, 0.446, 0.657, 1.155, 0.753, 0.674, 0.562, 0.926, 0.596

4.2 Scale up simulation

To be added.

5 Further improvement

In Stan code:

- Need a data matrix X,
- Need matrix multiplication,

In data:

- Need a driver index $d = 1, 2, \cdots, K$ for each shift k
- Need a data matrix X