# Sir roi rac version2

lephongplus2016

July 2020

## 1 Introduction

### 1.1 Trường hợp rời rạc

Trường hợp rời rạc của mô hình SIR là hệ phương trình sai phân sau:

$$R(n+1) = R(n) + \gamma(\Delta t)I(n)$$
$$I(n+1) = I(n) - \gamma(\Delta t)I(n) + \beta(\Delta t)I(n)S(n)$$
$$S(n+1) = S(n) - \beta(\Delta t)S(n)I(n)$$

Với $S_0 > 0$,$I_0 > 0$,$R_0 > 0$, ta có thỏa mãn tính chất sau: $S_0 + I_0 + R_0 = N$

- $S(n)$ : Số người có nguy cơ mắc bệnh;
- $I(n)$ : Số người bị nhiễm bệnh;
- $R(n)$ : Số người hồi phục sau bệnh;
- $\beta(n)$ : Tỷ lệ tiếp xúc của người trong nhóm $S(t)$ với người trong nhóm $I(n)$, được tính dựa trên các yếu tố thực tế như bán kính đi lại, mức độ tiếp xúc giữa người mắc bệnh với người có khả năng mắc bệnh, tính chất của virus,...;
- $\gamma(n)$ : Tỷ lệ hồi phục khi mắc bệnh, được tính bằng cách lấy 1 chia cho khoảng thời gian trung bình mà một người phục hồi;
- $N(n)$ : Tổng số người trong cộng đồng cách ly;

Giả sử rằng $\beta$ , $\gamma$ là các giá trị không đổi và có thể xác định từ các dữ liệu ban đầu. Các cá nhân phục hồi từ căn bệnh sẽ trở miễn dịch vĩnh viễn.
Trong hệ rời rạc chỉ xét số nguyên dương cho n=1,2,... .

Ta có hệ các phương trình thể hiện tốc độ thay đổi $S$, $I$, $R$:

$$R^{'} = \gamma(\Delta t)I$$
$$I^{'} = -\gamma(\Delta t)I + \beta(\Delta t)IS$$
$$S^{'} = -\beta(\Delta t)SI$$

Giới hạn của $S$ thì phụ thuộc vào các điều kiện ban đầu
Nếu $S_0 \leq \frac{\gamma}{\beta}$, thì $I_1 \leq I_0$ và $S_n$ giảm dần $I_{n+1} \leq I_n$ hay $I^{'} < 0$; Không còn dịch bệnh.
Ngược lại, nếu $S_0 > \frac{\gamma}{\beta}$, thì $I_1 > I_0$ hay $I^{'} > 0$ nhóm người bị nhiễm bệnh đang gia tăng.

## 2 Bài toán 3

Nhóm chọn ngôn ngữ thống kê R để thực hiện bài toán.

Mô tả cách thiết lập và chạy Metropolis-Hastings dựa trên Markov chain Monte Carlo (MCMC) trong phạm vi mô hình SIR.
Ví dụ code sử dụng dữ liệu Eyam.

---

```
library(MultiBD)

data(Eyam)

print(Eyam)

printsample_size <- 177
```

---

Tính log của phân bố xác suất số ca mắc bệnh và số người có khả năng mắc bệnh với param cho trước. Ma trận xác suất được trả về bởi $ddb\_prob()$ tương ứng với giá trị $S$ từ $a$ tới $a_0$ và $I$ từ $0$ tới $B$.
$a$ là cận dưới, $B$ là cận trên.
$ddb\_prob()$ Function là Xác suất chuyển tiếp của một quá trình chết/ sinh - chết..
`drates1` Tỉ lệ chết đi của $S$, bằng 0 vì $S$ chỉ chuyển số lượng sang cho $I$.
`brates2` Mô hình không sinh thêm phần tử.
`drates2` Tỉ lệ chết của $I$ là có và bằng `alpha*b`.
`trans12` Ở đây là mô hình chuyển từ trạng thái $S$ sang trạng thái $I$

---

```
loglik_sir <- function(param, data) {

  alpha <- exp(param[1]) # Rates must be non-negative
  beta <- exp(param[2])
  # Set-up SIR model
  drates1 <- function(a, b) { 0 }
  brates2 <- function(a, b) { 0 }
  drates2 <- function(a, b) { alpha * b }
  trans12 <- function(a, b) { beta * a * b }
  sum(sapply(1:(nrow(data) - 1), # Sum across all time steps k
             function(k) {
               log( # sums the logs of logs that change from T to T + 1 state
                 dbd_prob( # Compute the transition probability matrix
                   t = data$time[k + 1] - data$time[k], # Time increment
                   a0 = data$S[k], b0 = data$I[k], # From: S(t_k), I(t_k)
                   drates1, brates2, drates2, trans12,
                   a = data$S[k + 1], B = data$S[k] + data$I[k] - data$S[k + 1],
                   computeMode = 4, nblocks = 80 # Compute using 4 threads
                 )[1, data$I[k + 1] + 1] # To: S(t_(k+1)), I(t_(k+1))
               )
             }))
}
```

Phân bố xác suất tiên nghiệm của số ca mắc bệnh và số ca phục hồi dựa trên phân bố chuẩn.

```
logprior <- function(param) {
  log_alpha <- param[1]
  log_beta <- param[2]
  dnorm(log_alpha, mean = 0, sd = 100, log = TRUE) +
    dnorm(log_beta, mean = 0, sd = 100, log = TRUE)
}

library(MCMCpack)

alpha0 <- 3.39
beta0 <- 0.0212

post_sample <- MCMCmetrop1R(fun = function(param) { loglik_sir(param, Eyam) +
    logprior(param) },
                            theta.init = log(c(alpha0, beta0)),
                            mcmc = 177, burnin = 50)
# Sampling of beta, gamm
```

Vẽ đồ thị minh họa:

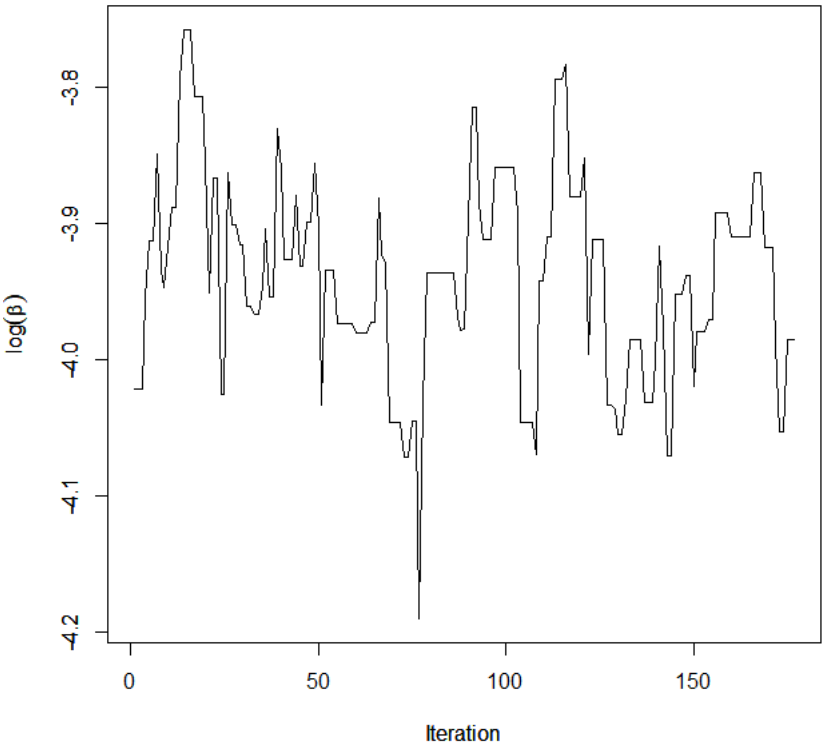
```
plot(as.vector(post_sample[,1]), type = "l", xlab = "Iteration", ylab =
    expression(log(alpha)))

plot(as.vector(post_sample[,2]), type = "l", xlab = "Iteration", ylab =
    expression(log(beta)))


R_0 <- 0
for(i in 1:8){
   R_0 <- R_0 + exp(loglik_sir(c(post_sample[i,1], post_sample[i,2]),
       Eyam))*(exp(post_sample[i,1])/exp(post_sample[i,2]))
}

print(R_0)
```

Kết quả tính được với ví dụ trên là : `The Metropolis acceptance rate was 0.51542`
$R_0$ là 1.543797e-15.




Hình 1: Đồ thị minh họa