元器件生存时间模拟分析-代码

Miao Cai

2019-04-09

# 定义函数

set.seed(123)  
  
alpha = matrix(c(0.001, 0.05, 0.0001, 0.08), ncol = 2, byrow = TRUE)  
w = c(45, 55)  
t = matrix(seq(10, 40, 10), ncol = 2, byrow = TRUE)  
K = c(10, 50, 100)  
  
  
lambda = function(r = 1:2, j = 1:2) {  
 alpha\_r0 = alpha[r, 1]  
 alpha\_r1 = alpha[r, 2]  
 return(alpha\_r0\*exp(alpha\_r1\*w[j]))  
}  
  
p = function(i = 1:2, j = 1:2){  
 lam1j = lambda(1, j) # failure due to factor 1  
 lam2j = lambda(2, j) # failure due to factor 2  
   
 p0 = exp(-(lam1j + lam2j)\*t[i, j])  
 p1 = (lam1j/(lam1j + lam2j))\*(1 - p0)  
 p2 = (lam2j/(lam1j + lam2j))\*(1 - p0)  
   
 return(c(p0, p1, p2))  
}  
  
SimMatrix = function(N = 10, N\_device = 10){  
 T1 = matrix(rep(-1, 3\*N), ncol = 3, byrow = TRUE)  
 T2 = T1; T3 = T1; T4 = T1;  
 T1 = t(rmultinom(N, N\_device, prob = p(1, 1)))  
 for (i in 1:N) {  
 T2[i,] = t(rmultinom(1, T1[i,1], prob = p(1, 2)))  
 }  
 for (i in 1:N) {  
 T3[i,] = t(rmultinom(1, T2[i,1], prob = p(2, 1)))  
 }  
 for (i in 1:N) {  
 T4[i,] = t(rmultinom(1, T3[i,1], prob = p(2, 2)))  
 }  
 return(list(T1, T2, T3, T4))  
}  
  
DAT0 = SimMatrix(100, 100)  
  
  
create\_standat = function(N = 10, N\_device = 10){  
 dat\_list = SimMatrix(N, N\_device)  
 stan\_dat = list(  
 n = nrow(dat\_list[[1]]),  
 DAT1 = dat\_list[[1]],  
 DAT2 = dat\_list[[2]],  
 DAT3 = dat\_list[[3]],  
 DAT4 = dat\_list[[4]],  
 W = w,  
 Tim = t  
 )  
 return(stan\_dat)  
}  
  
post\_result = function(N\_sim = 10, N = 10, N\_device = 10){  
 tab\_result = matrix(rep(NA, 4\*N\_sim), byrow = TRUE, ncol = 4)  
   
 for (i in 1:N\_sim) {  
 stan\_dat = create\_standat(N, N\_device)  
 fit00 <- stan(  
 model\_code = reliabMAC, data = stan\_dat,   
 warmup = 500, iter = 1000, chains = 1, cores = 1, seed = i)  
 tab\_result[i,] = summary(fit00)$summary[1:4,1]  
 }  
   
 return(tab\_result)  
}

# Stan代码

library(rstan)  
rstan\_options(auto\_write = TRUE)  
options(scipen = 99)  
  
reliabMAC = "  
data {  
 int n;  
 int DAT1[n, 3];  
 int DAT2[n, 3];  
 int DAT3[n, 3];  
 int DAT4[n, 3];  
 vector[2] W;  
 matrix[2, 2] Tim;  
}  
parameters{  
 real<lower=0, upper=0.1> a10;   
 real<lower=0, upper=0.1> a11;  
 real<lower=0, upper=0.1> a20;   
 real<lower=0, upper=0.1> a21;  
}  
transformed parameters{  
 simplex[3] p11;  
 simplex[3] p12;  
 simplex[3] p21;  
 simplex[3] p22;  
   
 p11[1] = exp(-(a10\*exp(a11\*W[1]) + a20\*exp(a21\*W[1]))\*Tim[1, 1]);  
 p11[2] = (a10\*exp(a11\*W[1]))/(a10\*exp(a11\*W[1]) + a20\*exp(a21\*W[1]))\*(1 - p11[1]);  
 p11[3] = 1-p11[1]-p11[2];  
   
 p12[1] = exp(-(a10\*exp(a11\*W[2]) + a20\*exp(a21\*W[2]))\*Tim[1, 2]);  
 p12[2] = (a10\*exp(a11\*W[2]))/(a10\*exp(a11\*W[2]) + a20\*exp(a21\*W[2]))\*(1 - p12[1]);  
 p12[3] = 1-p12[1]-p12[2];  
   
 p21[1] = exp(-(a10\*exp(a11\*W[1]) + a20\*exp(a21\*W[1]))\*Tim[2, 1]);  
 p21[2] = (a10\*exp(a11\*W[1]))/(a10\*exp(a11\*W[1]) + a20\*exp(a21\*W[1]))\*(1 - p21[1]);  
 p21[3] = 1-p21[1]-p21[2];  
   
 p22[1] = exp(-(a10\*exp(a11\*W[2]) + a20\*exp(a21\*W[2]))\*Tim[2, 2]);  
 p22[2] = (a10\*exp(a11\*W[2]))/(a10\*exp(a11\*W[2]) + a20\*exp(a21\*W[2]))\*(1 - p22[1]);  
 p22[3] = 1-p22[1]-p22[2];  
 //p12 = [p0, p1, p2]';  
 //p21 = [p0, p1, p2]';  
 //p22 = [p0, p1, p2]';  
}  
model{  
 for (i in 1:n){  
 //target += multinomial\_lpmf(DAT[i,] | p);  
 DAT1[i,] ~ multinomial(p11);  
 //DAT[i,] ~ multi\_log(p0, p1, p2);  
 }  
 for (i in 1:n){  
 DAT2[i,] ~ multinomial(p12);  
 }  
 for (i in 1:n){  
 DAT3[i,] ~ multinomial(p21);  
 }  
 for (i in 1:n){  
 DAT4[i,] ~ multinomial(p22);  
 }  
 a10 ~ uniform(0, 0.1);  
 a11 ~ uniform(0, 0.1);  
 a20 ~ uniform(0, 0.1);  
 a21 ~ uniform(0, 0.1);  
}  
"

# A1

set.seed(123)  
library(data.table)  
alpha = matrix(c(0.001, 0.05, 0.0001, 0.08), ncol = 2, byrow = TRUE)  
  
A1\_10 = post\_result(100, 1, 10)  
A1\_50 = post\_result(100, 1, 50)  
A1\_100 = post\_result(100, 1, 100)  
  
colMeans(A1\_10)  
colMeans(A1\_50)  
colMeans(A1\_100)  
  
apply(A1\_10, 2, sd)  
apply(A1\_50, 2, sd)  
apply(A1\_100, 2, sd)  
  
data.table::fwrite(as.data.frame(A1\_10), 'data/N = 10/A1\_10.csv')  
data.table::fwrite(as.data.frame(A1\_50), 'data/N = 10/A1\_50.csv')  
data.table::fwrite(as.data.frame(A1\_100), 'data/N = 10/A1\_100.csv')

## A2

set.seed(123)  
alpha = matrix(c(0.005, 0.05, 0.0005, 0.08), ncol = 2, byrow = TRUE)  
  
A2\_10 = post\_result(100, 10, 10)  
A2\_50 = post\_result(100, 10, 50)  
A2\_100 = post\_result(100, 10, 100)  
  
colMeans(A2\_10)  
colMeans(A2\_50)  
colMeans(A2\_100)  
  
apply(A2\_10, 2, sd)  
apply(A2\_50, 2, sd)  
apply(A2\_100, 2, sd)  
  
data.table::fwrite(as.data.frame(A2\_10), 'data/N = 10/A2\_10.csv')  
data.table::fwrite(as.data.frame(A2\_50), 'data/N = 10/A2\_50.csv')  
data.table::fwrite(as.data.frame(A2\_100), 'data/N = 10/A2\_100.csv')

# A3

set.seed(123)  
alpha = matrix(c(0.001, 0.05, 0.0001, 0.08), ncol = 2, byrow = TRUE)  
  
A3\_10 = post\_result(100, 10, 10)  
A3\_50 = post\_result(100, 10, 50)  
A3\_100 = post\_result(100, 10, 100)  
  
colMeans(A3\_10)  
colMeans(A3\_50)  
colMeans(A3\_100)  
  
apply(A3\_10, 2, sd)  
apply(A3\_50, 2, sd)  
apply(A3\_100, 2, sd)  
  
data.table::fwrite(as.data.frame(A3\_10), 'data/N = 10/A3\_10.csv')  
data.table::fwrite(as.data.frame(A3\_50), 'data/N = 10/A3\_50.csv')  
data.table::fwrite(as.data.frame(A3\_100), 'data/N = 10/A3\_100.csv')