

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO  
DEPARTAMENTO DE ESTATÍSTICA  
PROCESSOS ESTOCÁSTICOS

**Aplicação de Métodos MCMC**

Discente: Vitória Sesana

Doscente: Dr. Mauro Campos

Vitória - ES  
2025

# Sumário

<b>1</b>	<b>Algoritmo Mantel-Hastings</b>	<b>2</b>
<b>2</b>	<b>Amostrador de gibbs</b>	<b>3</b>
2.1	Questão 1: Função de Verossimilhança . . . . .	3
2.2	Questão 2: Distribuição a Posteriori . . . . .	3
2.3	Questão 3: Distribuições Condicionais Completas . . . . .	3
2.4	Estimativas . . . . .	4
<b>3</b>	<b>Anexo I</b>	<b>5</b>
3.1	Código Algoritmo MH com distribuição proposta Exponencial . . . . .	5
3.2	Código Algoritmo MH com distribuição proposta Log-normal . . . . .	7

# 1 Algoritmo Mantel-Hastings

O algoritmo de Metropolis-Hastings foi implementado para obter uma amostra de tamanho  $n = 10000$  da distribuição de Rayleigh quando  $\sigma = 4$ .

Para este cenário, as distribuições exponencial e log-normal foram propostas para o algoritmo.

O código está disponível no anexo.

No gráfico1, percebemos que a amostra via MH usando a distribuição exponencial como distribuição proposta não se adequa bem, principalmente se for comparado quando se utiliza a distribuição Log-Normal. Por conta disso, o resultado do segundo algoritmo será o escolhido para analisar as estimativas.

Figura 1: Gráfico de densidade das distribuições propostas para gerar valores de Rayleigh.

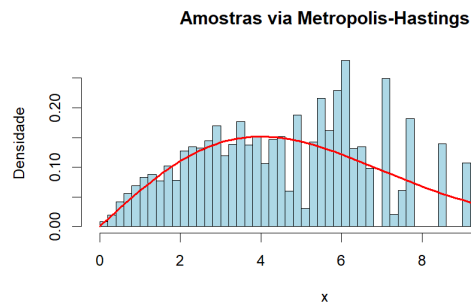


Figura 2: Gráfico 1

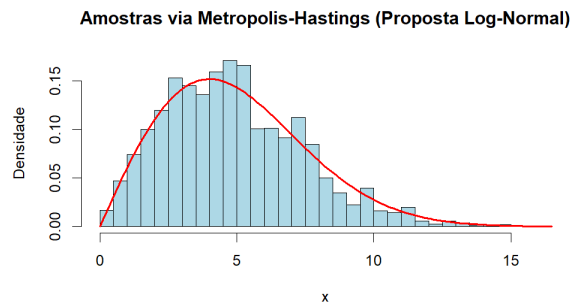


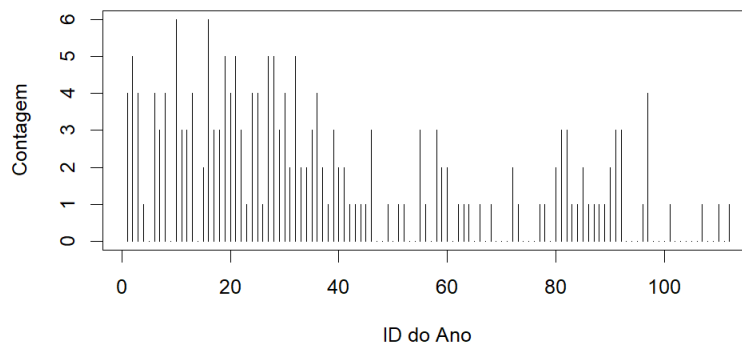
Figura 3: Gráfico 2

Na tabela 1, percebemos que as estimativas não aproximam tanto do valor teórico, apesar de no gráfico apresentado dar a ideia de um bom ajuste.

Tabela 1: Comparação dos valores observados e teóricos para as amostras geradas a partir da distribuição proposta Log-Normal.

	Valor teórico	Média amostral
Média	6.87	4.91
Variância	5.01	6.61

Figura 4: Plot dos dados em contagem.



## 2 Amostrador de gibbs

### 2.1 Questão 1: Função de Verossimilhança

Seja  $y_i$  o número de acidentes no ano  $i$ , para  $i = 1, \dots, 112$ . Suponha que haja um ponto de mudança  $k \in \{1, \dots, 112\}$  tal que:

$$\begin{cases} y_i \sim \text{Poisson}(\lambda), & \text{para } i = 1, \dots, k \\ y_i \sim \text{Poisson}(\nu), & \text{para } i = k+1, \dots, 112 \end{cases}$$

A função de verossimilhança conjunta dos dados, condicional a  $\lambda$ ,  $\nu$  e  $k$ , é dada por:

$$\mathcal{L}(\lambda, \nu, k \mid y) = \prod_{i=1}^k \frac{\lambda^{y_i} e^{-\lambda}}{y_i!} \cdot \prod_{i=k+1}^n \frac{\nu^{y_i} e^{-\nu}}{y_i!}$$

Desprezando os termos constantes em relação aos parâmetros, temos a verossimilhança proporcional a:

$$\mathcal{L}(\lambda, \nu, k \mid y) \propto \lambda^{\sum_{i=1}^k y_i} e^{-k\lambda} \cdot \nu^{\sum_{i=k+1}^n y_i} e^{-(n-k)\nu}$$

### 2.2 Questão 2: Distribuição a Posteriori

Assumindo as seguintes distribuições a priori independentes:

$$\lambda \sim \text{Gamma}(\alpha, \beta), \quad \nu \sim \text{Gamma}(\gamma, \delta), \quad k \sim \text{Uniforme Discreta}\{1, 2, \dots, n\}$$

A distribuição a posteriori conjunta dos parâmetros é proporcional ao produto da verossimilhança pelas distribuições a priori:

$$p(\lambda, \nu, k \mid y) \propto \mathcal{L}(\lambda, \nu, k \mid y) \cdot p(\lambda) \cdot p(\nu) \cdot p(k)$$

Substituindo as expressões:

$$p(\lambda, \nu, k \mid y) \propto \lambda^{\sum_{i=1}^k y_i + \alpha - 1} e^{-(k+\beta)\lambda} \cdot \nu^{\sum_{i=k+1}^n y_i + \gamma - 1} e^{-((n-k)+\delta)\nu}$$

### 2.3 Questão 3: Distribuições Condicionais Completas

As distribuições completas condicionais, necessárias para a implementação de um amostrador de Gibbs, são:

a) Para  $\lambda$ , condicional a  $k$  e  $y$ :

$$\lambda \mid \nu, k, y \sim \text{Gamma}\left(\alpha + \sum_{i=1}^k y_i, \beta + k\right)$$

b) Para  $\nu$ , condicional a  $k$  e  $y$ :

$$\nu \mid \lambda, k, y \sim \text{Gamma}\left(\gamma + \sum_{i=k+1}^n y_i, \delta + (n-k)\right)$$

c) Para  $k$ , condicional a  $\lambda$ ,  $\nu$  e  $y$  (discreta):

$$p(k \mid \lambda, \nu, y) \propto \lambda^{\sum_{i=1}^k y_i} e^{-k\lambda} \cdot \nu^{\sum_{i=k+1}^n y_i} e^{-(n-k)\nu}$$

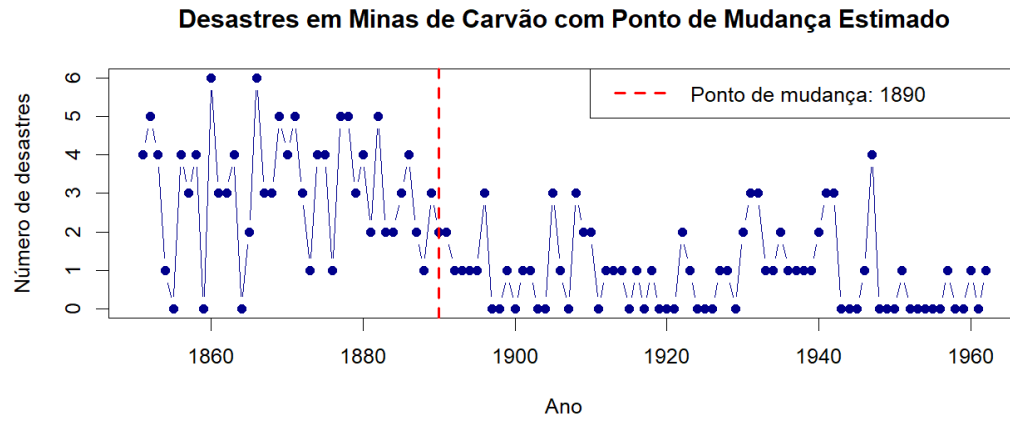
para  $k = 1, 2, \dots, n-1$ .

## 2.4 Estimativas

Tabela 2: Estimativas pontuais e intervalares de  $\lambda$ ,  $\nu$  e  $k$ .

	Estimativa pontual	2.5%	97.5%
$\lambda$	3.06	2.52	3.63
$\nu$	0.92	0.71	1.16
$k$	40.00	36.00	46.00

Figura 5: Gráfico de ponto de mundaça estimado.



## 3 Anexo I

### 3.1 Código Algoritmo MH com distribuição proposta Exponencial

```
1 # aplica o rayleight -----
2
3 # distribui o alvo = rayleight
4 # distribui o proposta = exponencial
5
6 # Fun o densidade da distribui o Rayleigh com sigma = 4
7 rayleigh_target <- function(x, sigma = 4) {
8   ifelse(x < 0, 0, (x / sigma^2) * exp(-x^2 / (2 * sigma^2)))
9 }
10
11 # Fun o densidade da Exponencial(1)
12 exp_proposal <- function(x, lambda = 1) {
13   ifelse(x < 0, 0, lambda * exp(-lambda * x))
14 }
15
16 # Algoritmo de Metropolis-Hastings
17 set.seed(123) # para reprodutibilidade
18 n_samples <- 10000
19 samples <- numeric(n_samples)
20 samples[1] <- rexp(1) # ponto inicial
21
22 for (i in 2:n_samples) {
23   current_x <- samples[i - 1]
24   proposed_x <- rexp(1) # proposta a partir de Exponencial(1)
25
26   # C lculo da raz o de aceita o
27   A <- (rayleigh_target(proposed_x) / rayleigh_target(current_x)) *
28     (exp_proposal(current_x) / exp_proposal(proposed_x))
29
30   # Aceita ou rejeita
31   if (runif(1) < A) {
32     samples[i] <- proposed_x
33   } else {
34     samples[i] <- current_x
35   }
36 }
37
38 # Visualiza o
39 hist(samples, breaks = 50, probability = TRUE, col = "lightblue",
40       main = "Amostras via Metropolis-Hastings",
41       xlab = "x", ylab = "Densidade")
42 curve((x / 16) * exp(-x^2 / (2 * 16)), add = TRUE, col = "red", lwd = 2) # Densidade Rayleigh(
43   =4)
44
45 # # comparando -----
46 #
47 # library(VGAM)
48 #
49 # # Fun o densidade da distribui o Rayleigh com sigma = 4
50 # rayleigh_target <- function(x, sigma = 4) {
51 #   ifelse(x < 0, 0, (x / sigma^2) * exp(-x^2 / (2 * sigma^2)))
52 # }
53 #
54 # # Fun o densidade da Exponencial(1)
55 # exp_proposal <- function(x, lambda = 1) {
56 #   ifelse(x < 0, 0, lambda * exp(-lambda * x))
57 # }
58 #
59 # # Algoritmo de Metropolis-Hastings
60 # set.seed(123)
61 # n_samples <- 10000
62 # samples <- numeric(n_samples)
63 # samples[1] <- rexp(1)
64 #
65 # for (i in 2:n_samples) {
66 #   current_x <- samples[i - 1]
67 #   proposed_x <- rexp(1)
68 # }
```

```

69 #   A <- (rayleigh_target(proposed_x) / rayleigh_target(current_x)) *
70 #       (exp_proposal(current_x) / exp_proposal(proposed_x))
71 #
72 #   if (runif(1) < A) {
73 #       samples[i] <- proposed_x
74 #   } else {
75 #       samples[i] <- current_x
76 #   }
77 # }
78 #
79 # # Gráfico: histograma + densidade teórica com dRayleigh
80 # hist(samples, breaks = 50, probability = TRUE, col = "lightblue",
81 #       main = "Amostras Metropolis-Hastings vs Rayleigh( =4)",
82 #       xlab = "x", xlim = c(0, max(samples)))
83 #
84 # # Curva da densidade Rayleigh( =4) com dRayleigh do pacote VGAM
85 # curve(drayleigh(x, scale = 4), add = TRUE, col = "red", lwd = 2)
86 #
87 # legend("topright", legend = c("Densidade teórica Rayleigh( =4)"),
88 #        col = c("red"), lwd = 2)
89 #
90 # samples
91 #
92 # mean(samples)
93 # sd(samples)^2
94 #
95 #
96 # # gamma -----
97 #
98 # # comparando -----
99 #
100 # library(VGAM)
101 #
102 # # Função da densidade da distribuição Rayleigh com sigma = 4
103 # rayleigh_target <- function(x, sigma = 4) {
104 #   ifelse(x < 0, 0, (x / sigma^2) * exp(-x^2 / (2 * sigma^2)))
105 # }
106 #
107 # # Função da densidade da Exponencial(1)
108 # exp_proposal <- function(x, lambda = 1) {
109 #   ifelse(x < 0, 0, lambda * exp(-lambda * x))
110 # }
111 #
112 # # Algoritmo de Metropolis-Hastings
113 # set.seed(123)
114 # n_samples <- 10000
115 # samples <- rpois(n_samples, lambda = 6)
116 # samples[1] <- rexp(1)
117 #
118 # for (i in 2:n_samples) {
119 #   current_x <- samples[i - 1]
120 #   proposed_x <- rexp(1)
121 #
122 #   A <- (rayleigh_target(proposed_x) / rayleigh_target(current_x)) *
123 #       (exp_proposal(current_x) / exp_proposal(proposed_x))
124 #
125 #   if (runif(1) < A) {
126 #       samples[i] <- proposed_x
127 #   } else {
128 #       samples[i] <- current_x
129 #   }
130 # }
131 #
132 # # Gráfico: histograma + densidade teórica com dRayleigh
133 # hist(samples, breaks = 50, probability = TRUE, col = "lightblue",
134 #       main = "Amostras Metropolis-Hastings vs Rayleigh( =4)",
135 #       xlab = "x", xlim = c(0, max(samples)))
136 #
137 # # Curva da densidade Rayleigh( =4) com dRayleigh do pacote VGAM
138 # curve(drayleigh(x, scale = 4), add = TRUE, col = "red", lwd = 2)
139 #
140 # legend("topright", legend = c("Densidade teórica Rayleigh( =4)"),
141 #        col = c("red"), lwd = 2)

```

```

142 #
143 # samples
144 #
145 # mean(samples)
146 # sd(samples)^2
147 #

```

## 3.2 Código Algoritmo MH com distribuição proposta Log-normal

```

1 # -----
2 # Aplica o Rayleigh com proposta Log-Normal
3 # -----
4
5 # Distribui o alvo = Rayleigh
6 # Distribui o proposta = Log-Normal
7
8 # Fun o densidade da distribui o Rayleigh com sigma = 4
9 rayleigh_target <- function(x, sigma = 4) {
10   ifelse(x < 0, 0, (x / sigma^2) * exp(-x^2 / (2 * sigma^2)))
11 }
12
13 # Fun o densidade da Log-Normal
14 lognorm_proposal <- function(x, meanlog = 0, sdlog = 1) {
15   dlnorm(x, meanlog = meanlog, sdlog = sdlog)
16 }
17
18 # Par metros da proposta Log-Normal
19 meanlog <- 0
20 sdlog <- 1
21
22 # Algoritmo de Metropolis-Hastings
23 set.seed(123) # para reprodutibilidade
24 n_samples <- 10000
25 samples <- numeric(n_samples)
26 samples[1] <- rlnorm(1, meanlog = meanlog, sdlog = sdlog) # ponto inicial
27
28 for (i in 2:n_samples) {
29   current_x <- samples[i - 1]
30   proposed_x <- rlnorm(1, meanlog = meanlog, sdlog = sdlog) # proposta a partir de Log-Normal
31
32   # C lculo da raz o de aceita o
33   A <- (rayleigh_target(proposed_x) / rayleigh_target(current_x)) *
34     (lognorm_proposal(current_x, meanlog, sdlog) / lognorm_proposal(proposed_x, meanlog, sdlog))
35
36   # Aceita ou rejeita
37   if (runif(1) < A) {
38     samples[i] <- proposed_x
39   } else {
40     samples[i] <- current_x
41   }
42 }
43
44 # Visualiza o
45 hist(samples, breaks = 50, probability = TRUE, col = "lightblue",
46       main = "Amostras via Metropolis-Hastings (Proposta Log-Normal)",
47       xlab = "x",
48       ylab = "Densidade")
49 curve((x / 16) * exp(-x^2 / (2 * 16)), add = TRUE, col = "red", lwd = 2) # Densidade Rayleigh( =
50
51
52 media_obs <- mean(samples)
53 var_obs <- sd(samples)^2
54
55 # valor teorico
56
57 media_teorica <- ((4-pi)/2)*(4^2)
58 var_teorica <- 4*sqrt(pi/ 2)
59
60 tab1 <-
61   cbind(
62     media = c(media_teorica, media_obs),
63     var = c(var_teorica, var_obs)
64   ) %>% t() %>% round(4) %>%

```



```

65  as.data.frame()
66
67  colnames(tab1) <- c("Valor te rico", "M dia amostral")
68
69  tab1 %>%
70    xtable::xtable()

1  y <- data$Contagem
2  n <- length(y)
3
4  # -----
5  # 2. Hiperpar metros
6  # -----
7  alpha <- 1
8  beta <- 1
9  gamma <- 1
10 delta <- 1
11
12 n_iter <- 10000
13
14 # -----
15 # 3. Inicializa o
16 # -----
17 lambda_samples <- numeric(n_iter)
18 nu_samples <- numeric(n_iter)
19 k_samples <- numeric(n_iter)
20
21 lambda <- 1
22 nu <- 1
23 k <- floor(n / 2)
24
25 # -----
26 # 4. Gibbs Sampling
27 # -----
28 for (t in 1:n_iter) {
29   # Atualizar lambda
30   shape_lambda <- alpha + sum(y[1:k])
31   rate_lambda <- beta + k
32   lambda <- rgamma(1, shape_lambda, rate_lambda)
33
34   # Atualizar nu
35   shape_nu <- gamma + sum(y[(k+1):n])
36   rate_nu <- delta + (n - k)
37   nu <- rgamma(1, shape_nu, rate_nu)
38
39   # Atualizar k
40   log_probs <- numeric(n - 1)
41   for (ki in 1:(n - 1)) {
42     sum1 <- sum(y[1:ki])
43     sum2 <- sum(y[(ki + 1):n])
44     log_probs[ki] <- sum1 * log(lambda) - ki * lambda +
45       sum2 * log(nu) - (n - ki) * nu
46   }
47
48   probs <- exp(log_probs - max(log_probs)) # estabiliza o
49   probs <- probs / sum(probs)
50   k <- sample(1:(n - 1), 1, prob = probs)
51
52   # Armazenar
53   lambda_samples[t] <- lambda
54   nu_samples[t] <- nu
55   k_samples[t] <- k
56 }
57
58
59 # analises -----
60 # -----
61
62
63 # Estimativas pontuais
64 lambda_est <- mean(lambda_samples)
65 nu_est <- mean(nu_samples)
66 k_est <- round(mean(k_samples))

```

```

67
68 cat("Estimativas pontuais:\n")
69 cat(sprintf("    (antes): %.2f\n", lambda_est))
70 cat(sprintf("    (depois): %.2f\n", nu_est))
71 cat(sprintf("Ponto de mudan a estimado (k): %d (ano %d)\n", k_est, 1850 + k_est))
72
73 # Intervalos de credibilidade
74 ci_lambda <- quantile(lambda_samples, c(0.025, 0.975))
75 ci_nu <- quantile(nu_samples, c(0.025, 0.975))
76 ci_k <- quantile(k_samples, c(0.025, 0.975))
77
78 cat("\nIntervalos de credibilidade 95%:\n")
79 cat(sprintf("    : (%.2f, %.2f)\n", ci_lambda[1], ci_lambda[2]))
80 cat(sprintf("    : (%.2f, %.2f)\n", ci_nu[1], ci_nu[2]))
81 cat(sprintf("k: (%d, %d)      anos (%d, %d)\n",
82             ci_k[1], ci_k[2], 1850 + ci_k[1], 1850 + ci_k[2]))
83
84
85 plot(data$Ano, y, type = "b", pch = 16, col = "darkblue",
86       xlab = "Ano", ylab = "N mero de desastres",
87       main = "Desastres em Minas de Carv o com Ponto de Mudan a Estimado")
88 abline(v = 1850 + k_est, col = "red", lwd = 2, lty = 2)
89 legend("topright", legend = paste("Ponto de mudan a:", 1850 + k_est),
90       col = "red", lty = 2, lwd = 2)
91
92 estimativas <- rbind(lambda_est, nu_est, k_est) %>%
93   round(2)
94
95 ics <-
96   rbind(t(ci_lambda), t(ci_nu), t(ci_k)) %>%
97   round(2)
98
99 cbind(estimativas, ics) %>%
100   xtable::xtable()

```