

Algoritmo de Metropolis-Hastings por componentes

2026

Um vetor aleatório $(X_1, X_2)^\top$ tem função densidade

$$f(x_1, x_2) \propto \frac{(x_1 x_2)^5}{2^{x_1-1}} \left(1 - \frac{1}{2^{x_1}}\right)^{x_2-1} \exp(-(5x_1 + 7x_2)), \quad (1)$$

se $x_1 > 0$ e $x_2 > 0$; $f(x_1, x_2) = 0$, caso contrário.

O objetivo é gerar uma amostra de pares de $(X_1, X_2)^\top$.

A partir da expressão (1), as distribuições condicionais completas têm funções densidade tais que

$$f_1(x_1) = f(x_1|x_2) \propto \frac{x_1^5}{2^{x_1}} \left(1 - \frac{1}{2^{x_1}}\right)^{x_2-1} \exp(-5x_1) \quad (2)$$

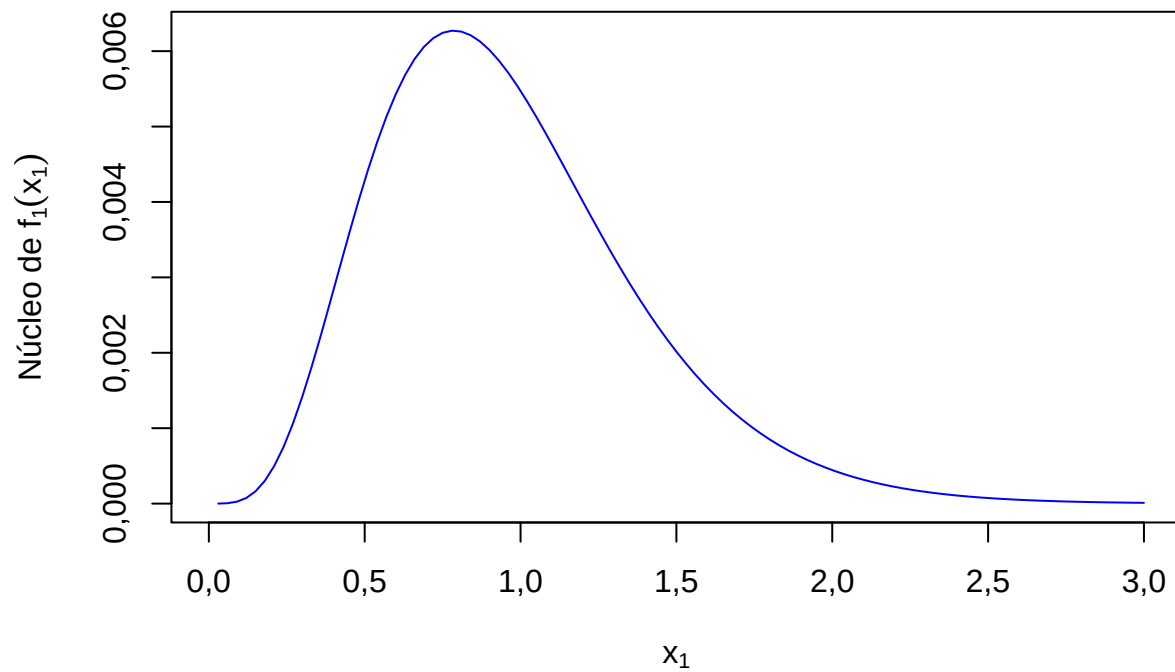
e

$$f_2(x_2) = f(x_2|x_1) \propto x_2^5 \left(1 - \frac{1}{2^{x_1}}\right)^{x_2} \exp(-7x_2). \quad (3)$$

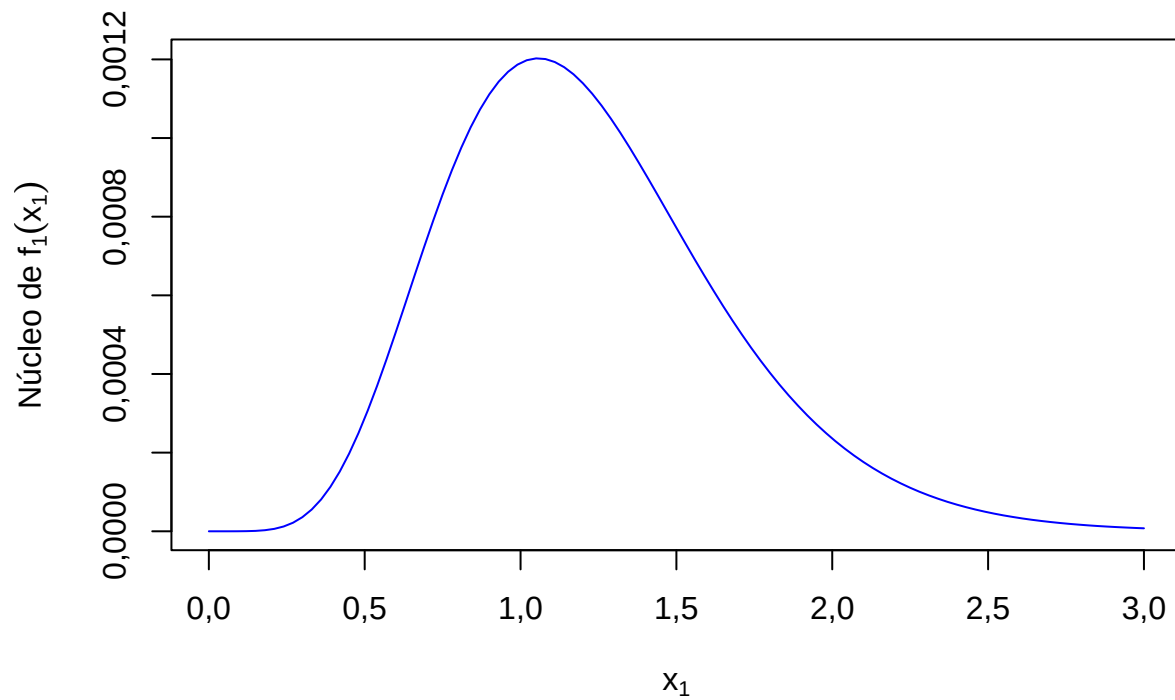
Os núcleos das funções em (2) e (3) não correspondem a distribuições conhecidas, pelo menos de imediato (mas vide a Nota 3, pag. 17). Os gráficos abaixo correspondem a diferentes valores de x_2 e x_1 .

```
# Separador decimal: ", "  
options(OutDec = ",")
```

```
# Núcleos das distribuições condicionais completas  
f1 <- function(x1, x2) {  
  return(x1^5 * (1 - 1 / 2^x1)^(x2 - 1) * exp(-5 * x1) / 2^x1)  
}  
  
curve(f1(x, x2 = 0.3), xlab = expression(x[1]), 0, 3, col = "blue",  
      ylab = expression(paste("Núcleo de ", f[1](x[1]))))
```

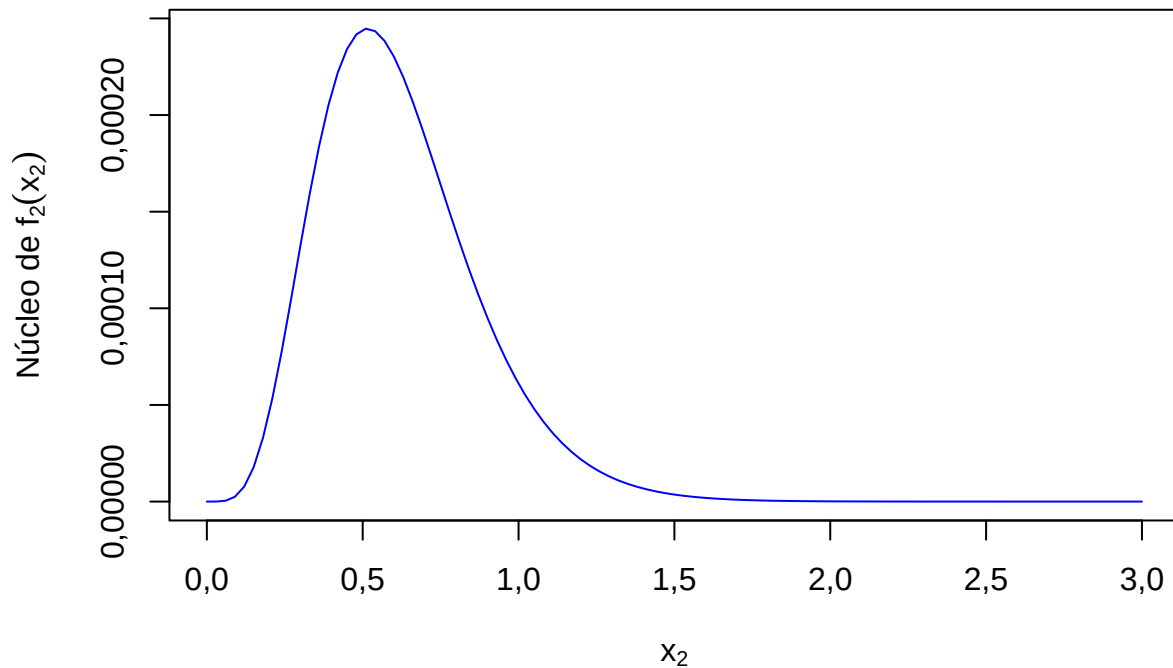


```
curve(f1(x, x2 = 2.5), xlab = expression(x[1]), 0, 3, col = "blue",
      ylab = expression(paste("Núcleo de ", f[1](x[1]))))
```

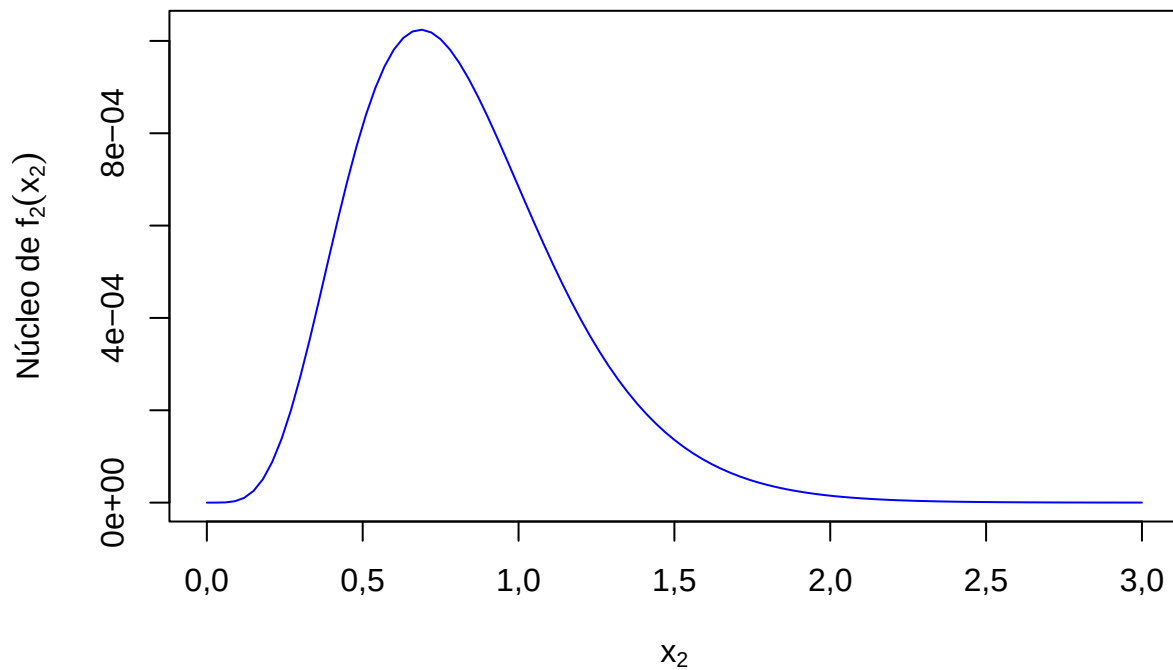


```
f2 <- function(x2, x1) {
  return(x2^5 * (1 - 1 / 2^x1)^x2 * exp(-7 * x2))
}

curve(f2(x, x1 = 0.1), xlab = expression(x[2]), 0, 3, col = "blue",
      ylab = expression(paste("Núcleo de ", f[2](x[2]))))
```



```
curve(f2(x, x1 = 2.0), xlab = expression(x[2]), 0, 3, col = "blue",
      ylab = expression(paste("Núcleo de ", f[2](x[2]))))
```



Utilizamos o algoritmo de Metropolis-Hastings por componentes. Tanto em (2) quanto em (3) adotamos uma distribuição proposta `gama(forma =  $ax$ , taxa =  $a$ )` com esperança $ax/a = x$ e variância $ax/a^2 = x/a$, sendo que a seleção do valor de a está relacionada com a taxa de aceitação dos candidatos gerados. Neste processo, a é um parâmetro de ajuste (*tuning parameter*).

```
# Distribuições propostas: gama(forma = a * x, taxa = a)
a10 <- 1 # X1
a20 <- 1 # X2
```

Iniciamos gerando duas cadeias com 15000 pares de $(X_1, X_2)^\top$ em cada cadeia.

```
ncad <- 2 # Número de cadeias
nsim <- 15000
x1 <- x2 <- matrix(NA, ncad, nsim)

# Valores iniciais
x1[1, 1] <- 0.9
x2[1, 1] <- 2.5
x1[2, 1] <- 3.0
x2[2, 1] <- 0.1

# Contadores de aceitações
nac1 <- nac2 <- numeric(ncad)

# Simulação
set.seed(120)
for (m in 1:ncad) {
  for (j in 2:nsim) {
    # candidato x1
    y1 <- rgamma(1, shape = a10 * x1[m, j - 1], rate = a10)

    # probabilidade x1
    alfa <- min(1, f1(y1, x2[m, j - 1]) * dgamma(x1[m, j - 1], a10 * y1,
      rate = a10) / (f1(x1[m, j - 1], x2[m, j - 1]) * dgamma(y1, a10 *
      x1[m, j - 1], rate = a10)))

    # geração x1
    if (rbinom(1, 1, alfa) == 1) {
      x1[m, j] <- y1
      nac1[m] <- nac1[m] + 1
    } else {
      x1[m, j] <- x1[m, j - 1]
    }

    # candidato x2
    y2 <- rgamma(1, shape = a20 * x2[m, j - 1], rate = a20)

    # probabilidade x2
    alfa <- min(1, f2(y2, x1[m, j]) * dgamma(x2[m, j - 1], a20 * y2,
      rate = a20) / (f2(x2[m, j - 1], x1[m, j]) * dgamma(y2, a20 *
      x2[m, j - 1], rate = a20)))

    # geração x2
    if (rbinom(1, 1, alfa) == 1) {
      x2[m, j] <- y2
      nac2[m] <- nac2[m] + 1
    } else {
      x2[m, j] <- x2[m, j - 1]
    }
  }
}

cat("\n Taxa de aceitação de X1 (%):", round(nac1 / (nsim - 1) * 100, 1))
```

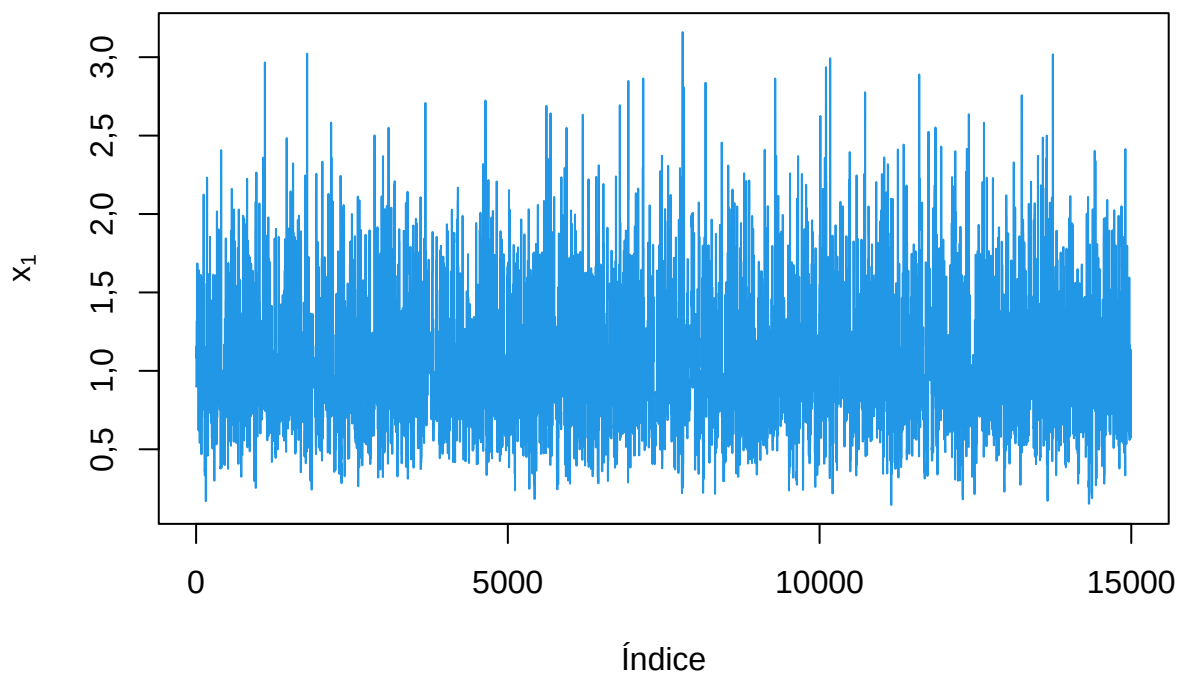
```
##
## Taxa de aceitação de X1 (%): 38,6 39,6
cat("\n Taxa de aceitação de X2 (%):", round(nac2 / (nsim - 1) * 100, 1))
```

```
##
## Taxa de aceitação de X2 (%): 33,4 33,5
```

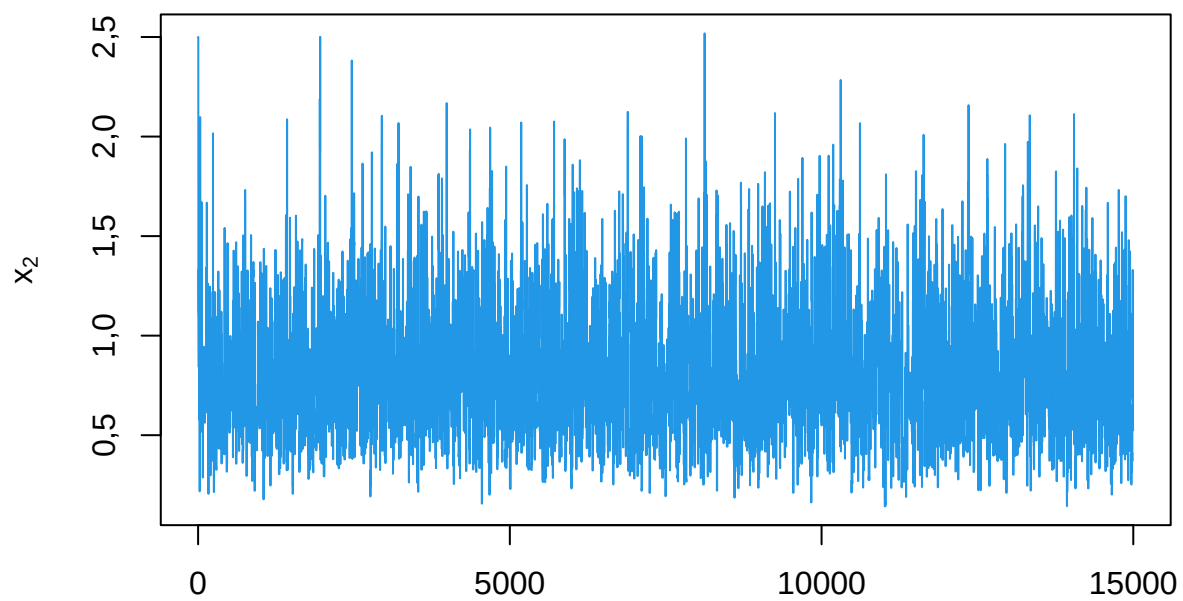
Em seguida as sequências geradas são apresentadas graficamente.

```
# Sequência
for (m in 1:ncad) {
  plot(x1[m,], type = "l", xlab = "Índice", ylab = expression(x[1]), col = 4,
       main = paste("Cadeia", m))
  plot(x2[m,], type = "l", xlab = "Índice", ylab = expression(x[2]), col = 4,
       main = paste("Cadeia", m))
}
```

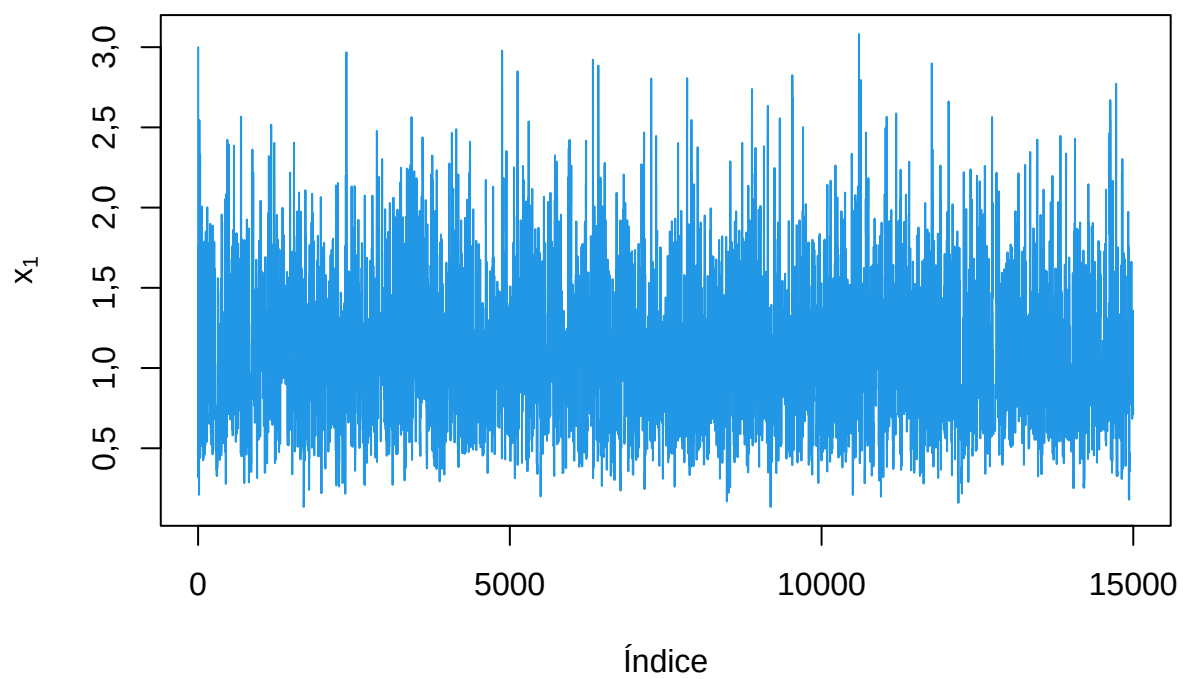
Cadeia 1



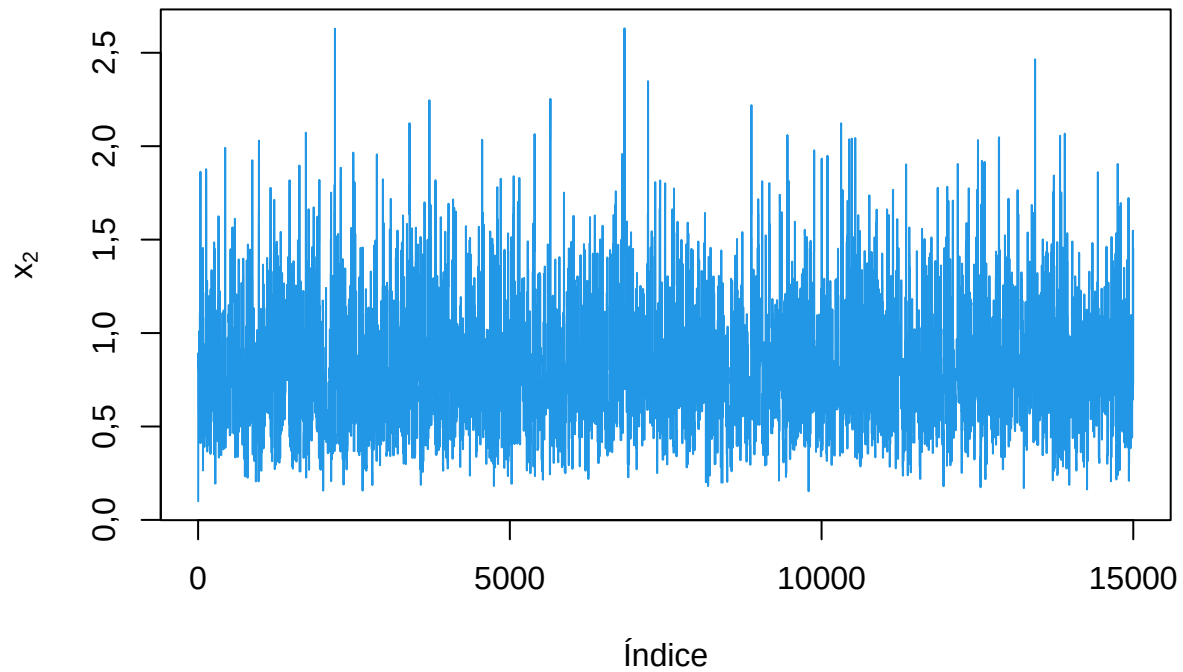
Cadeia 1



Cadeia 2

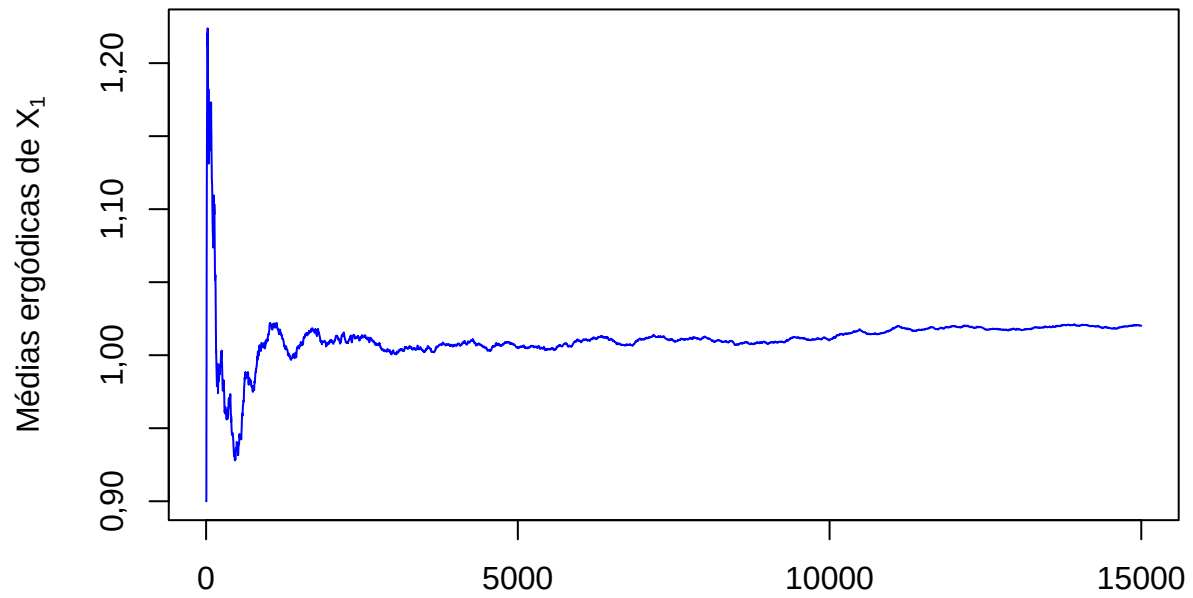


Cadeia 2

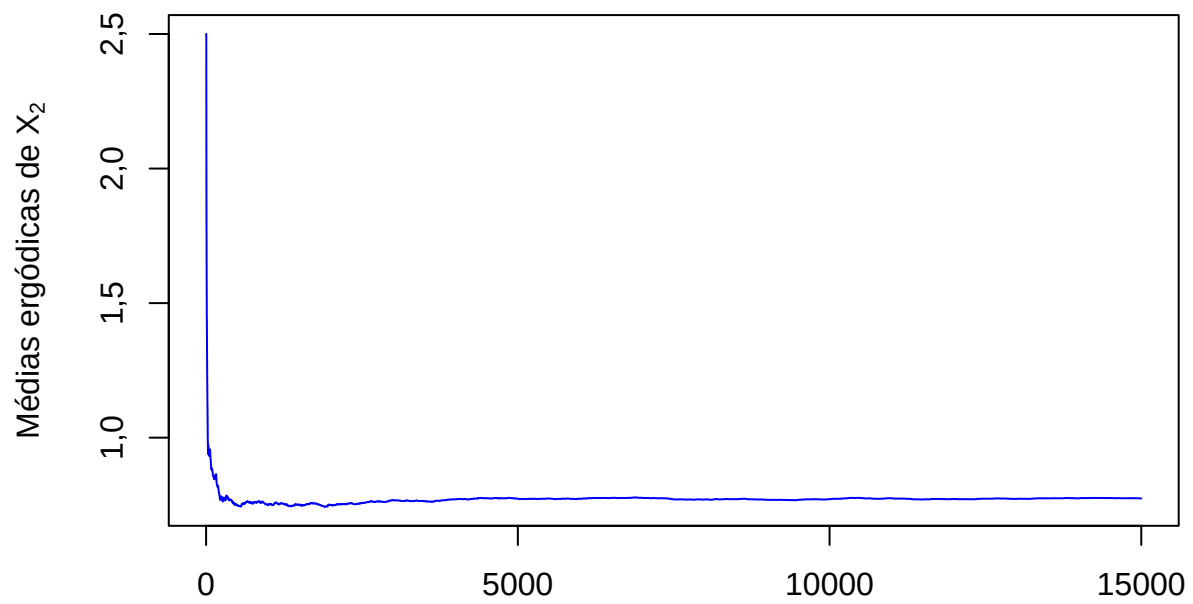


```
# Médias ergódicas
for (m in 1:ncad) {
  plot(cumsum(x1[m,]) / (1:nsim), type = "l", xlab = "Índice",
       ylab = expression(paste("Médias ergódicas de ", X[1])),
       main = paste("Cadeia", m), col = "blue")
  plot(cumsum(x2[m,]) / (1:nsim), type = "l", xlab = "Índice",
       ylab = expression(paste("Médias ergódicas de ", X[2])),
       main = paste("Cadeia", m), col = "blue")
}
```

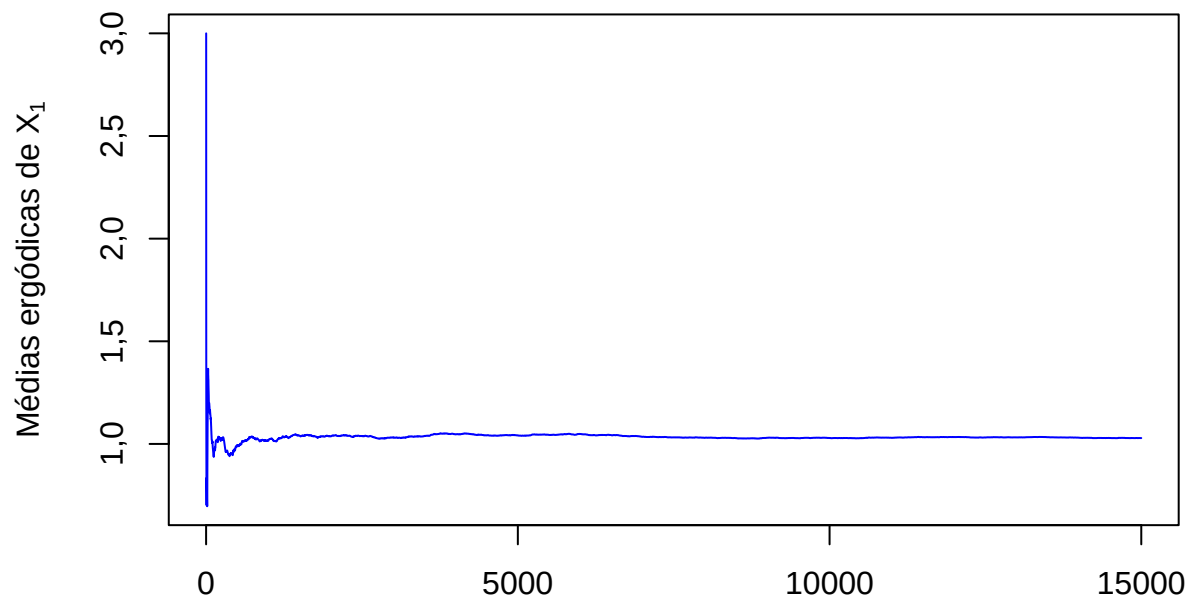
Cadeia 1



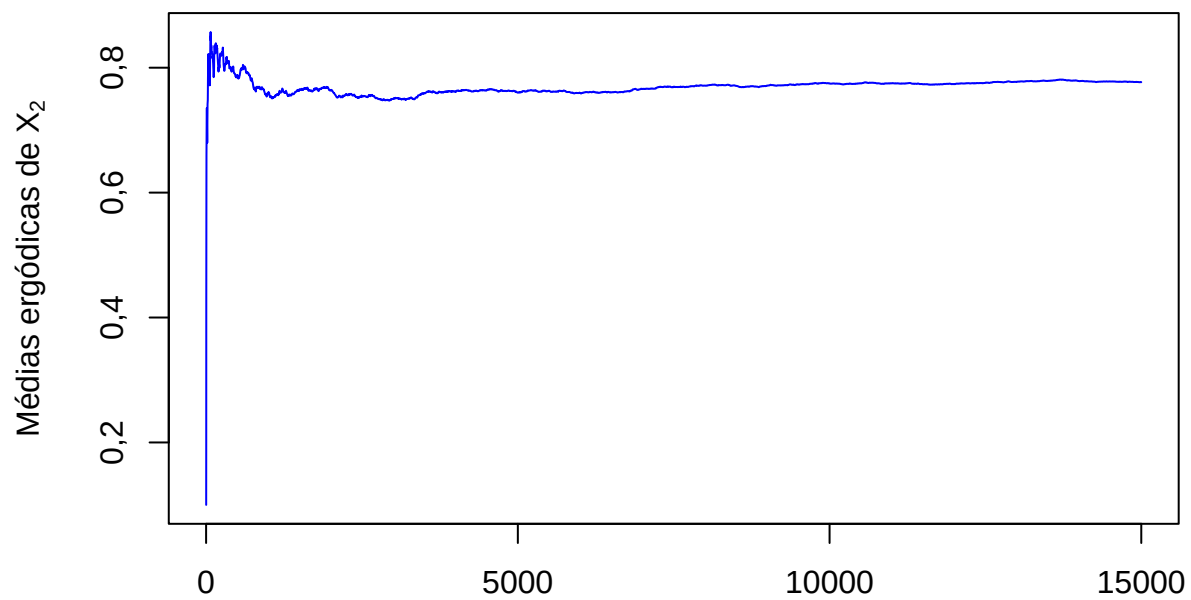
Índice Cadeia 1



Cadeia 2



Cadeia 2

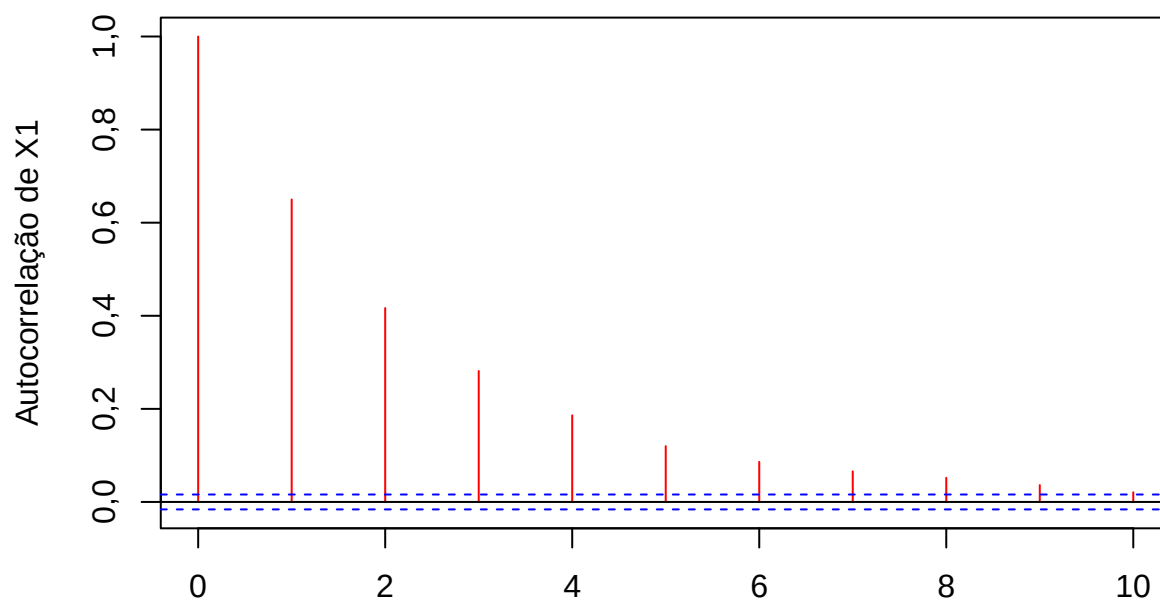


Índice

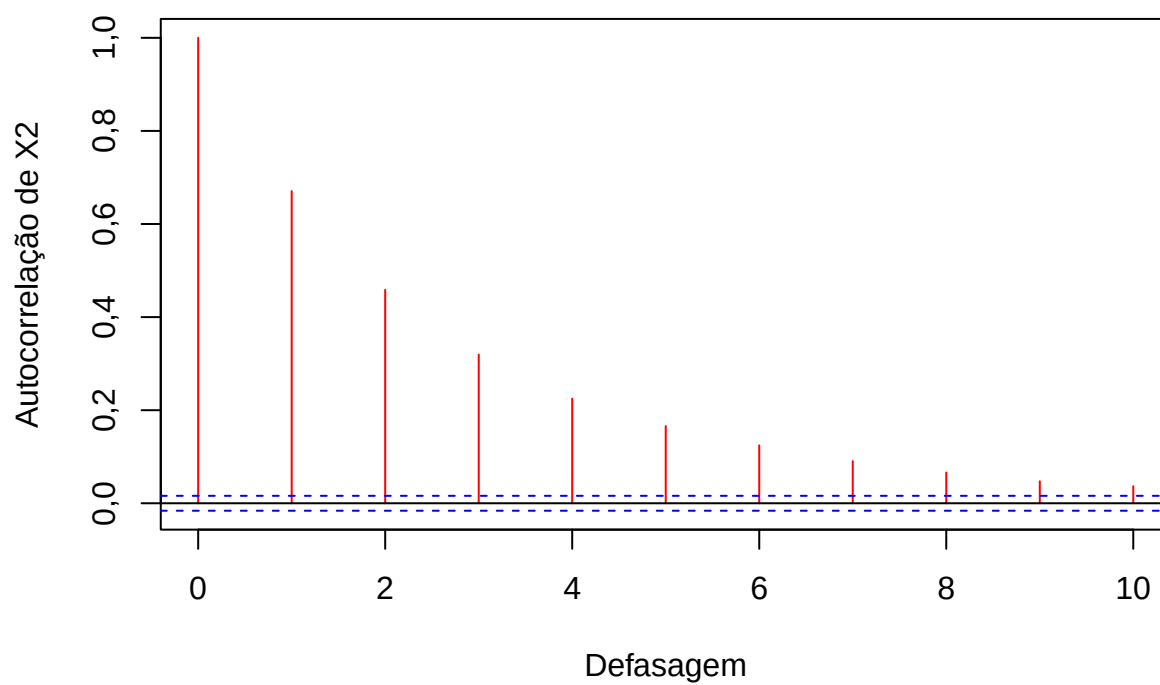
```
# Autocorrelação
for (m in 1:ncad) {
  acf(x1[m,], lag.max = 10, main = paste("Cadeia", m), xlab = "Defasagem",
    ylab = "Autocorrelação de X1", col = "red")
  acf(x2[m,], lag.max = 10, main = paste("Cadeia", m), xlab = "Defasagem",
    ylab = "Autocorrelação de X2", col = "red")
}
```

}

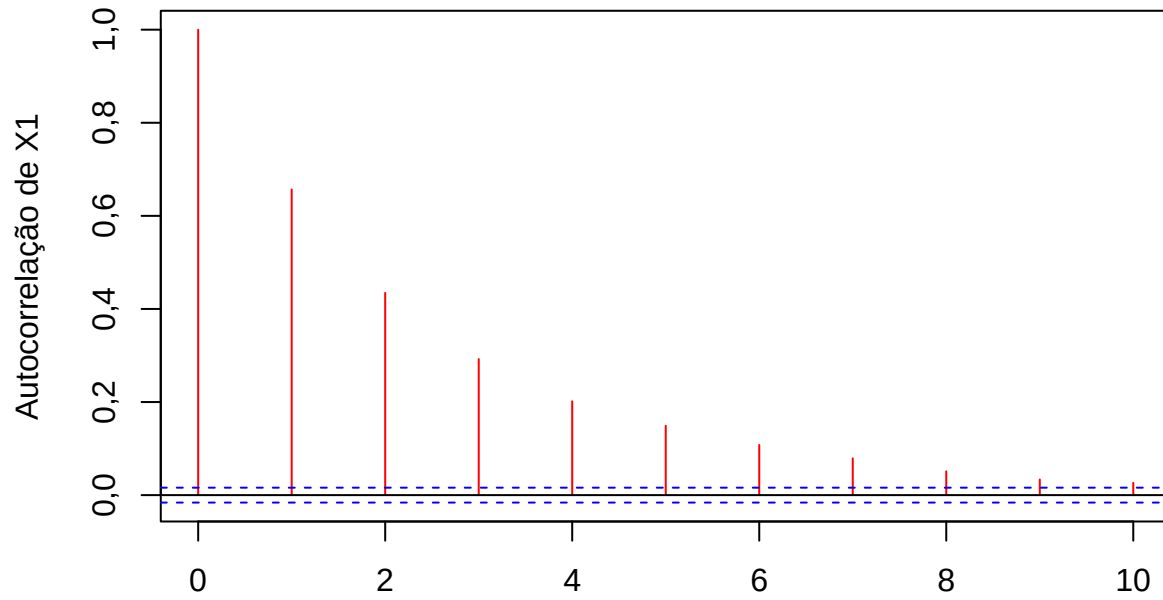
Cadeia 1



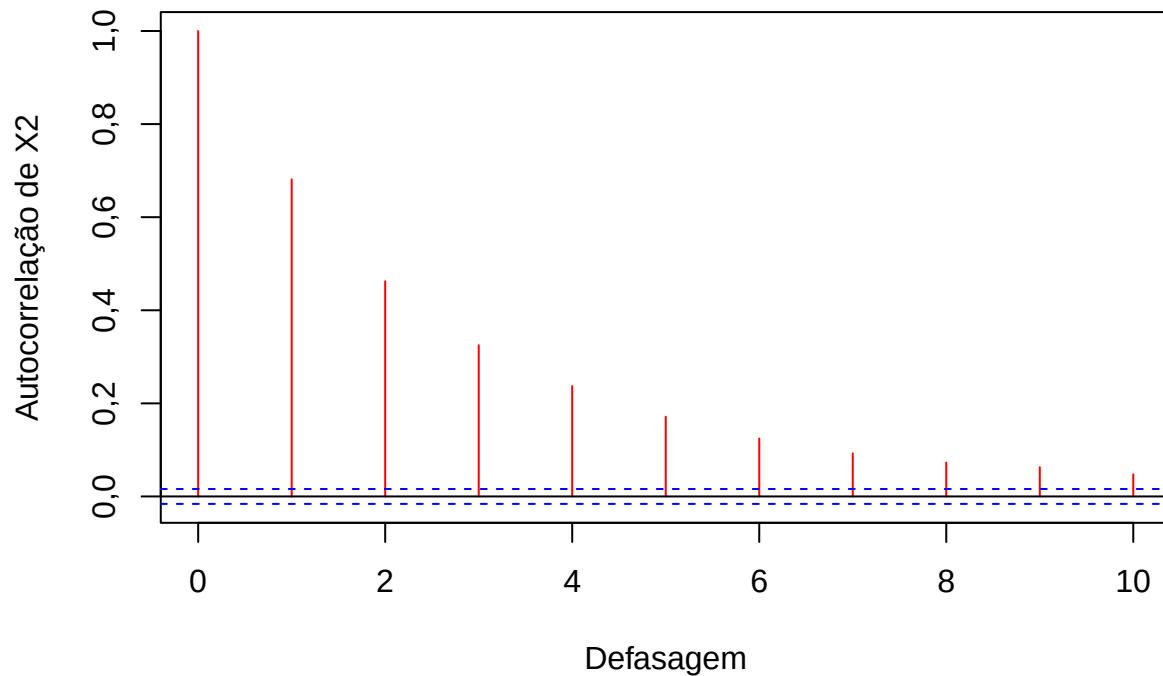
Cadeia 1



Cadeia 2



Cadeia 2



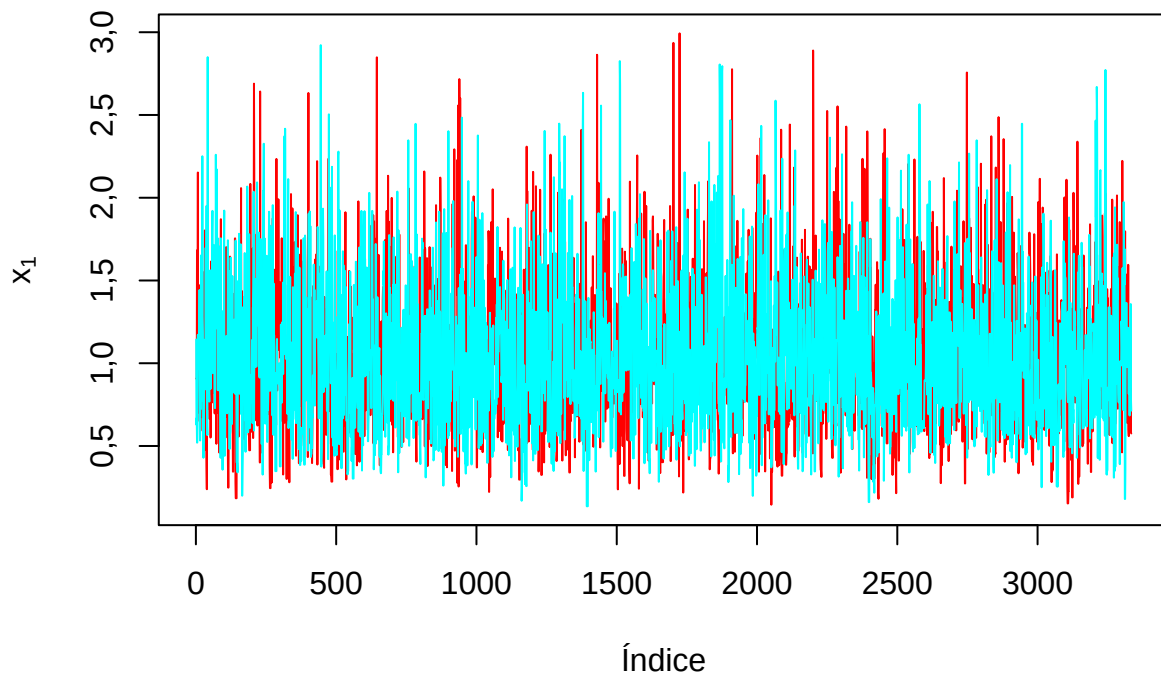
Apresentamos resultados para as sequências após levar em conta descarte (*burn-in*) e espaçamento (*thinning*). Nos gráficos das médias ergódicas notamos estabilização a partir de 5000 iterações. Nos gráficos de autocorrelações notamos que com três defasagens o valor máximo está em torno de 0,4.

```

# Amostras (descarte e espaçamento levados em conta)
descarte <- 5000 # burn-in
espac <- 3      # thinning
indices <- seq(descarte + 1, nsim, by = espac)
x1 <- x1[, indices]
x2 <- x2[, indices]

# Sequência
cores <- rainbow(ncad)
plot(x1[1,], type = "l", xlab = "Índice", ylab = expression(x[1]), col = cores[1],
     ylim = range(x1))
for (m in 2:ncad) {
  lines(x1[m,], type = "l", col = cores[m])
}

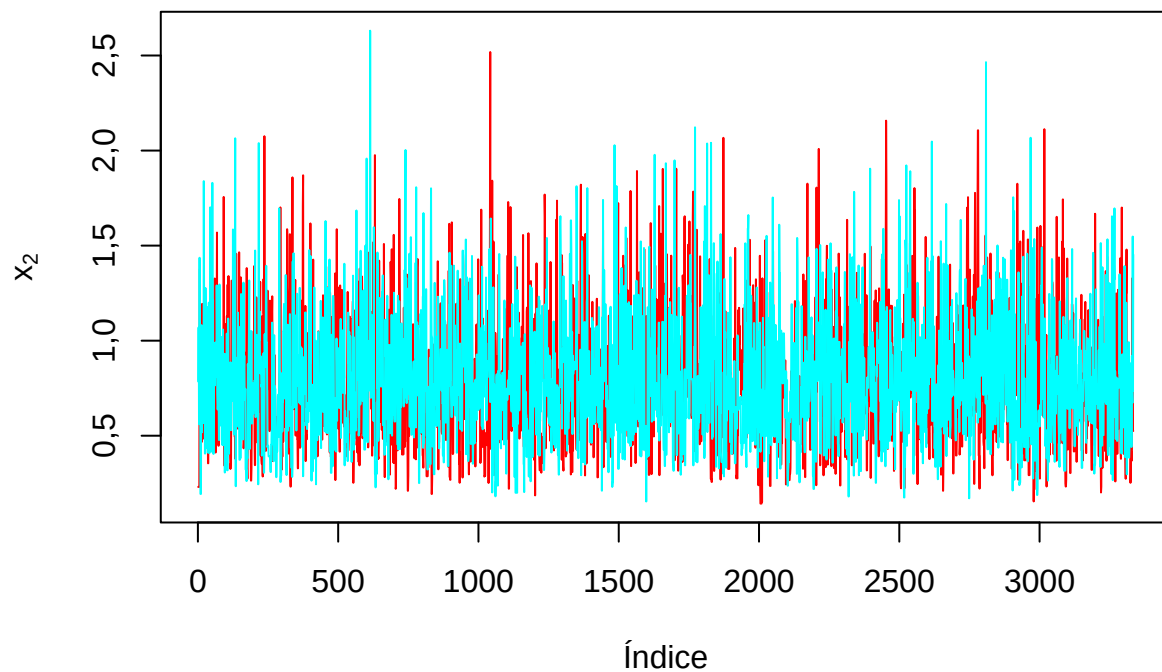
```



```

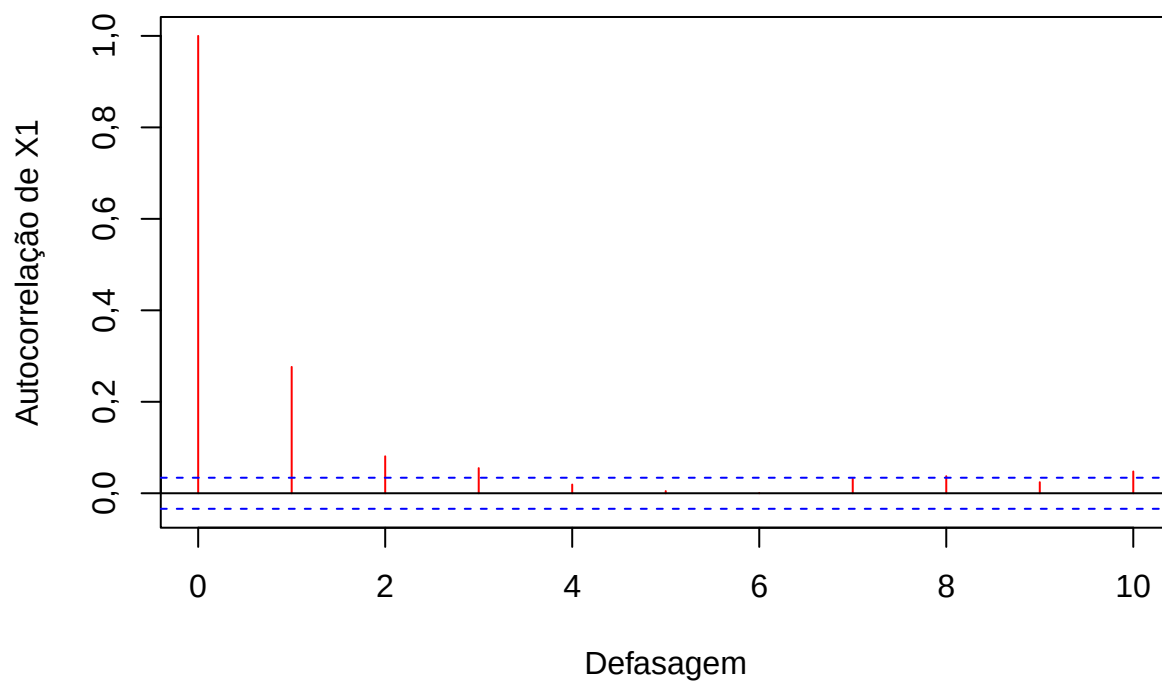
plot(x2[1,], type = "l", xlab = "Índice", ylab = expression(x[2]), col = cores[1],
     ylim = range(x2))
for (m in 2:ncad) {
  lines(x2[m,], type = "l", col = cores[m])
}

```

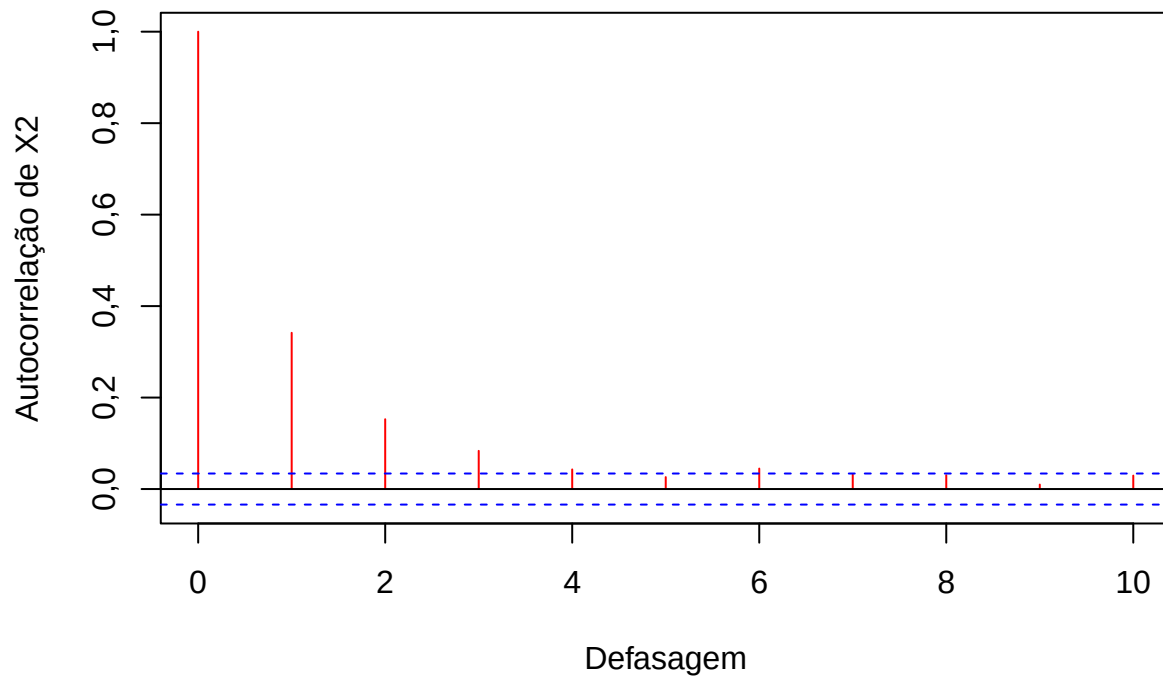


```
# Autocorrelação
for (m in 1:ncad) {
  acf(x1[m,], lag.max = 10, main = paste("Cadeia", m), xlab = "Defasagem",
    ylab = "Autocorrelação de X1", col = "red")
  acf(x2[m,], lag.max = 10, main = paste("Cadeia", m), xlab = "Defasagem",
    ylab = "Autocorrelação de X2", col = "red")
}
```

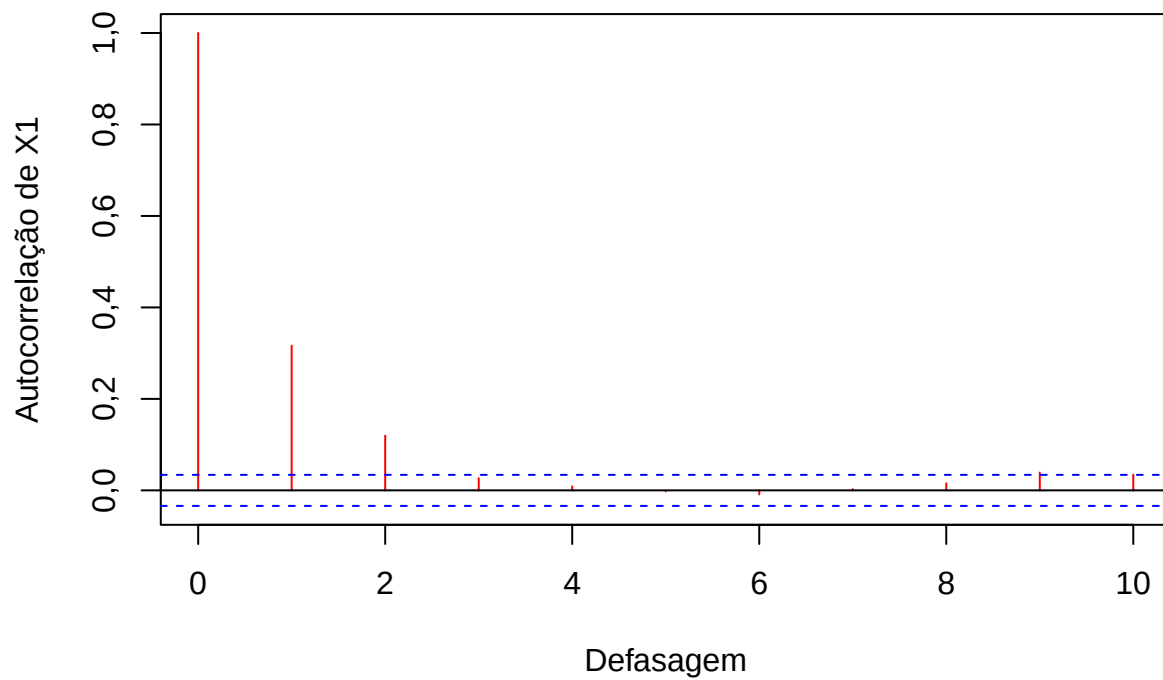
Cadeia 1



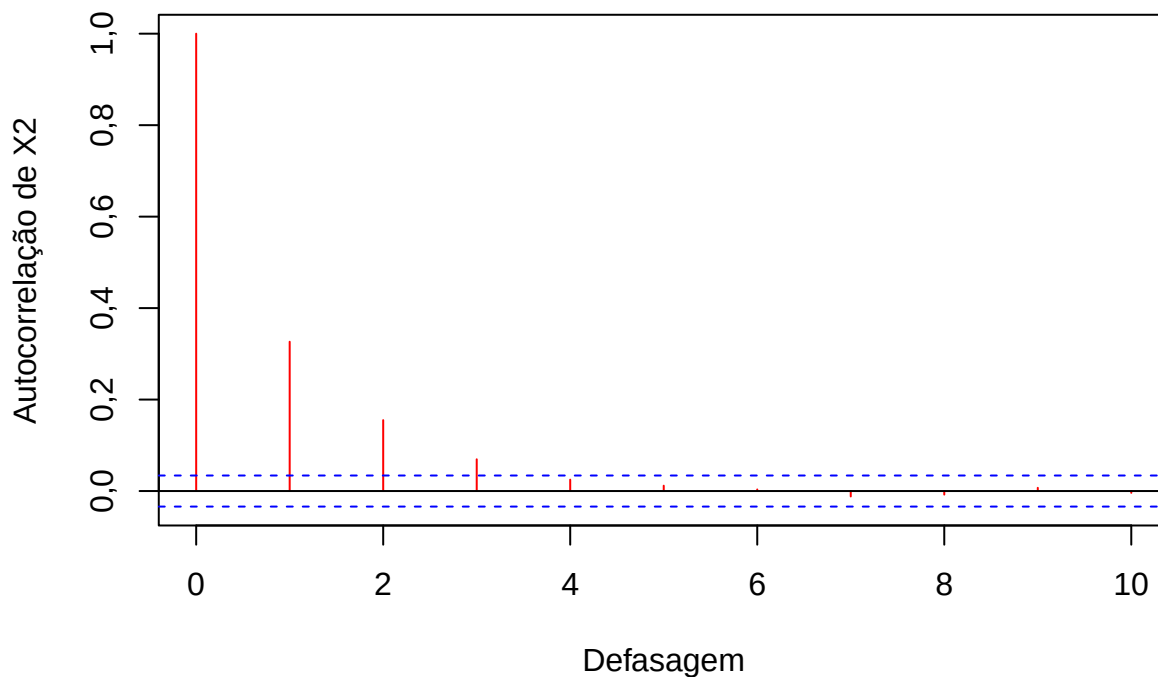
Cadeia 1



Cadeia 2



Cadeia 2



As linhas das matrizes geradas são combinadas para formar a amostra final.

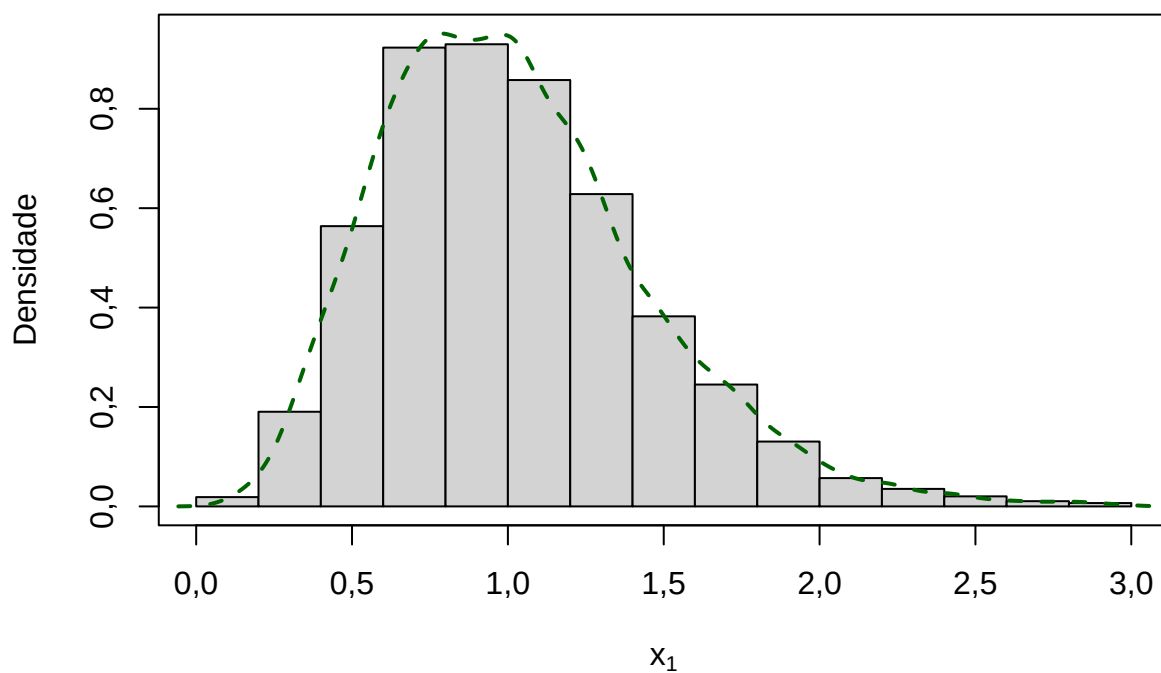
```
# Amostra final
x1 <- as.numeric(t(x1))
x2 <- as.numeric(t(x2))
for (nome in c("x1", "x2")) {
  cat("\n Estatísticas descritivas de", nome, "\n")
  print(summary(get(nome)))
}
```

```
##
## Estatísticas descritivas de x1
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0,1360 0,7099  0,9772  1,0249  1,2679  2,9932
##
## Estatísticas descritivas de x2
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0,1431 0,5528  0,7422  0,7802  0,9634  2,6306
cat("\n Tamanho da amostra: ", length(x1), "\n")
```

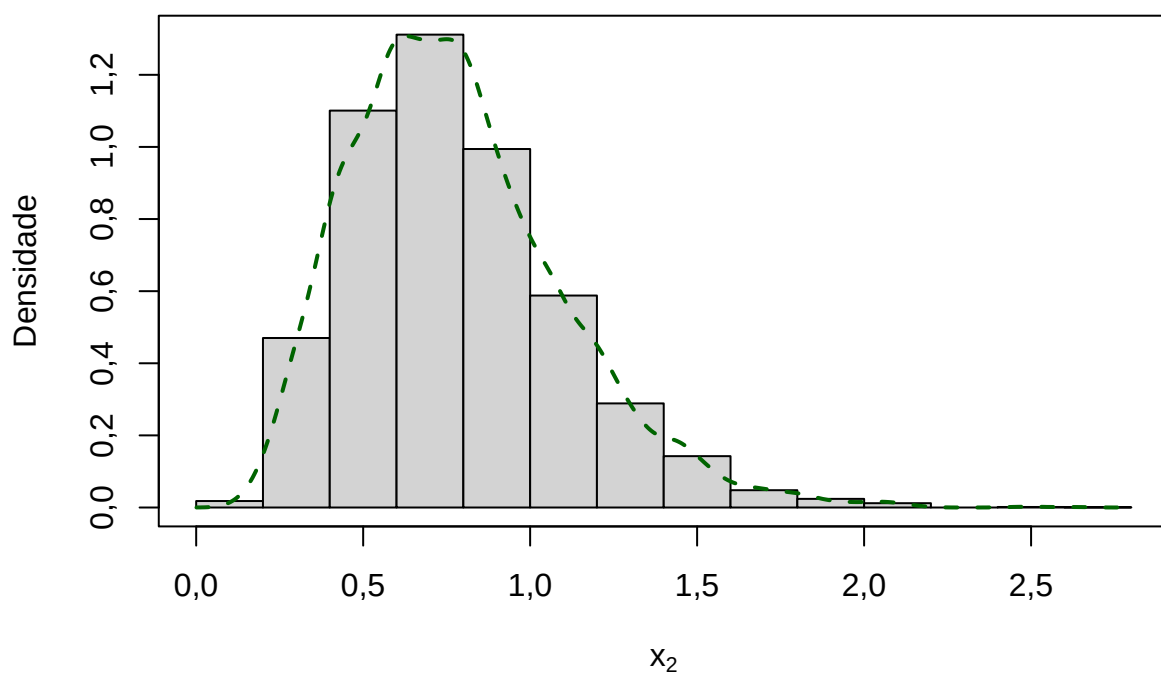
```
##
## Tamanho da amostra: 6668
```

```
# X1
fhist <- hist(x1, plot = FALSE)
fdens <- density(x1)
inty <- c(0, max(fhist$density, fdens$y))
plot(fhist, freq = FALSE, main = "", xlab = expression(x[1]),
     ylab = "Densidade", ylim = inty)
lines(fdens, lty = 2, col = "darkgreen", lwd = 2)
```

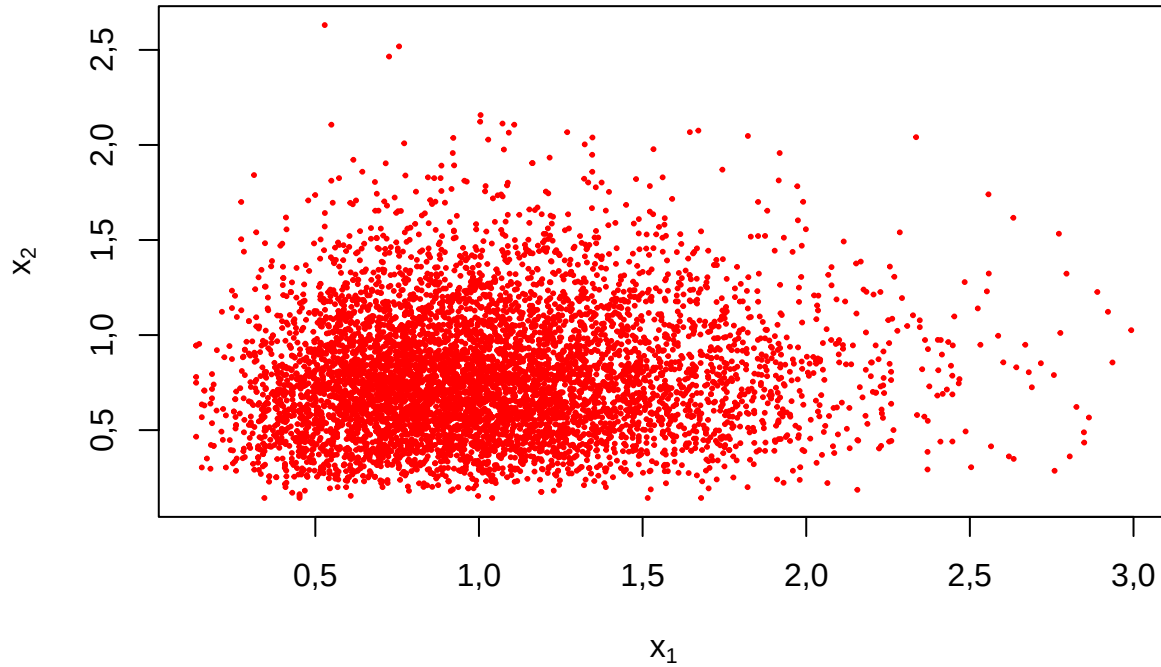
```
box()
```



```
# X2
fhist <- hist(x2, plot = FALSE)
fdens <- density(x2)
inty <- c(0, max(fhist$density, fdens$y))
plot(fhist, freq = FALSE, main = "", xlab = expression(x[2]),
     ylab = "Densidade", ylim = inty)
lines(fdens, lty = 2, col = "darkgreen", lwd = 2)
box()
```



```
plot(x1, x2, pch = 20, cex = 0.4, xlab = expression(x[1]),
     ylab = expression(x[2]), col = "red")
```



Nota 1 A amostra gerada de $(X_1, X_2)^\top$ pode ser usada, por exemplo, para obter uma estimativa de Monte Carlo de $P(X_1 > a_0, X_2 > b_0)$ em que $a_0 > 0$ e $b_0 > 0$ são fixados.

Nota 2 Para evitar que candidatos com valores muito próximos de 0 sejam gerados, os parâmetros de forma podem ser mudados para `shape = max(1.1, a10 * x1[m, j - 1])` e `shape = max(1.1, a20 * x2[m, j - 1])`, respectivamente.

Nota 3 Tomando $b_1 = 1 - 1/2^{x_1}$, o núcleo da função densidade na expressão (3) pode ser escrito como

$$\begin{aligned} x_2^5 b_1^{x_2} \exp(-7x_2) &= x_2^5 \exp(\log(b_1^{x_2})) \exp(-7x_2) \\ &= x_2^5 \exp(x_2 \log(b_1)) \exp(-7x_2) \\ &= x_2^5 \exp(-x_2 \{7 - \log(b_1)\}) \\ &= x_2^{6-1} \exp(-x_2 \{7 - \log(b_1)\}), \end{aligned}$$

de modo que a distribuição condicional completa de X_2 é gama com parâmetro de forma igual a 6 e parâmetro de taxa igual a

$$7 - \log(b_1) = 7 - \log\left(1 - \frac{1}{2^{x_1}}\right).$$

Refaça o exemplo utilizando este resultado. Neste caso, o esquema é híbrido, sendo que com X_1 aplicamos o algoritmo de Metropolis-Hastings ao passo que com X_2 aplicamos o amostrador de Gibbs.

Nota 4 Modifique os parâmetros das distribuições propostas (pag. 4) de modo a avaliar a sensibilidade dos resultados.