Algoritmo de Metropolis-Hastings por componentes

2023

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

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

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

Será apresentado um gerador de amostras do vetor $(X_1, X_2)^\top$ em linguagem R.

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

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

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

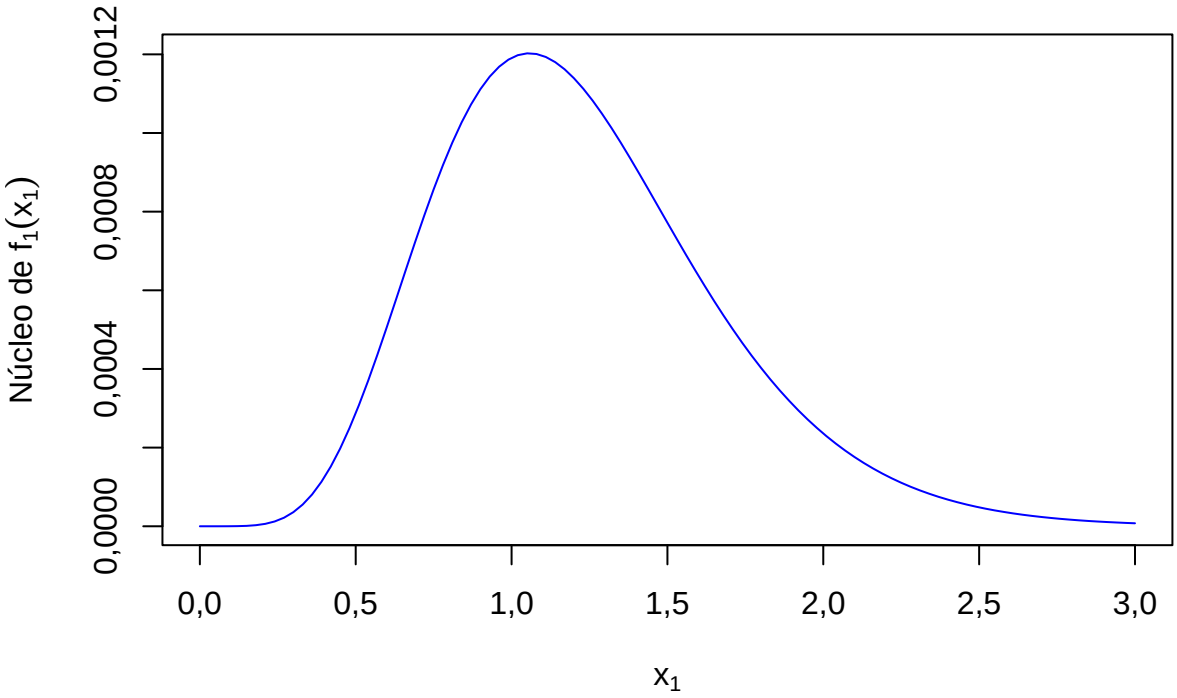
e

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

Os núcleos das funções em (2) e (3) não correspondem a distribuições conhecidas, pelo menos de imediato (mas vide a Nota 3, pag. 14). Os gráficos abaixo correspondem a $x_2 = 2,5$ e $x_1 = 1,5$, respectivamente.

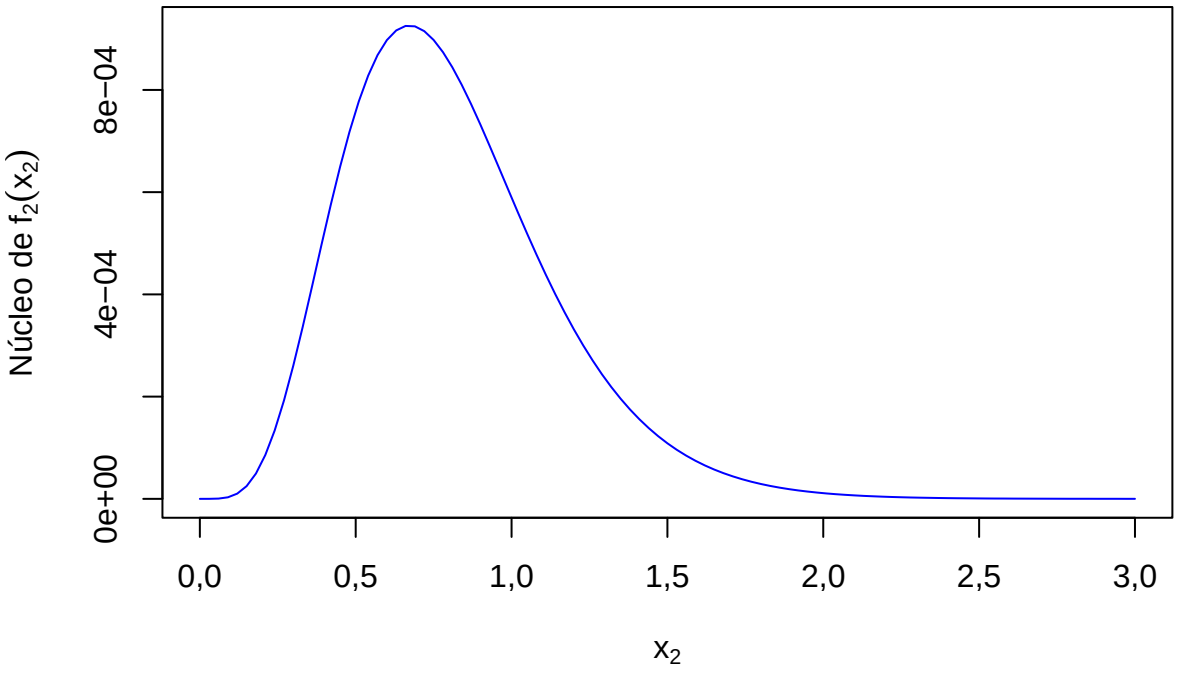
```
# Núcleos das distribuições condicionais completas
f1 <- function(x1, x2) {
   return(x1^5 * (1 - 1 / 2^x1)^(x2 - 1) * exp(-5 * x1) / 2^x1)
}

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




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

curve(f2(x, x1 = 1.5), 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(ax, 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.

```r
# Constantes para o amostrador de cada cadeia
set.seed(3316)
descarte <- 500
espac <- 1
M <- 3000
nsim <- descarte + espac * M
```

```r
# Distribuições propostas: gama(forma = a * x, taxa = a)
a10 <- 0.8 # X1
a20 <- 1   # X2
```

```r
## Cadeia 1
# Valores iniciais
x11 <- x21 <- c()
x11[1] <- 0.5
x21[1] <- 2.5

# Contadores de aceitações
nac1 <- nac2 <- 0

# Simulação
for (j in 2:nsim) {
   # candidato x1
   y1 <- rgamma(1, shape = a10 * x11[j - 1], rate = a10)

   # probabilidade x1
   alfa <- min(1, f1(y1, x21[j - 1]) * dgamma(x11[j - 1], a10 * y1,
      rate = a10) / (f1(x11[j - 1], x21[j - 1]) * dgamma(y1, a10 *
      x11[j - 1],  rate = a10)))

   # geração x1
   if (runif(1) > 1 - alfa) {
      x11[j] <- y1
      nac1 <- nac1 + 1
   } else {
      x11[j] <- x11[j - 1]
   }

   # candidato x2
   y2 <- rgamma(1, shape = a20 * x21[j - 1], rate = a20)

   # probabilidade x2
   alfa <- min(1, f2(y2, x11[j]) * dgamma(x21[j - 1], a20 * y2,
      rate = a20) / (f2(x21[j - 1], x11[j]) * dgamma(y2, a20 *
      x21[j - 1], rate = a20)))

   # geração x2
   if (runif(1) > 1 - alfa) {
      x21[j] <- y2
      nac2 <- nac2 + 1
   } else {
      x21[j] <- x21[j - 1]
   }
}
```

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

```
##
##  Taxa de aceitação de X1 (%): 36,6
```

```r
cat("\n Taxa de aceitação de X2 (%):", round(nac2 / (nsim - 1) * 100, 1))
```

```
##
##  Taxa de aceitação de X2 (%): 32,6
```

```r
## Cadeia 2
# Valores iniciais
x12 <- x22 <- c()
x12[1] <- 1.9
x22[1] <- 0.1

# Contadores de aceitações
nac1 <- nac2 <- 0

# Simulação
for (j in 2:nsim) {
   # candidato x1
   y1 <- rgamma(1, shape = a10 * x12[j - 1], rate = a10)

   # probabilidade x1
   alfa <- min(1, f1(y1, x22[j - 1]) * dgamma(x12[j - 1], a10 * y1,
      rate = a10) / (f1(x12[j - 1], x22[j - 1]) * dgamma(y1, a10 *
      x12[j - 1],  rate = a10)))

   # geração x1
   if (runif(1) > 1 - alfa) {
      x12[j] <- y1
      nac1 <- nac1 + 1
   } else {
      x12[j] <- x12[j - 1]
   }

   # candidato x2
   y2 <- rgamma(1, shape = a20 * x22[j - 1], rate = a20)

   # probabilidade x2
   alfa <- min(1, f2(y2, x12[j]) * dgamma(x22[j - 1], a20 * y2,
      rate = a20) / (f2(x22[j - 1], x12[j]) * dgamma(y2, a20 *
      x22[j - 1], rate = a20)))

   # geração x2
   if (runif(1) > 1 - alfa) {
      x22[j] <- y2
      nac2 <- nac2 + 1
   } else {
      x22[j] <- x22[j - 1]
   }
}
```

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

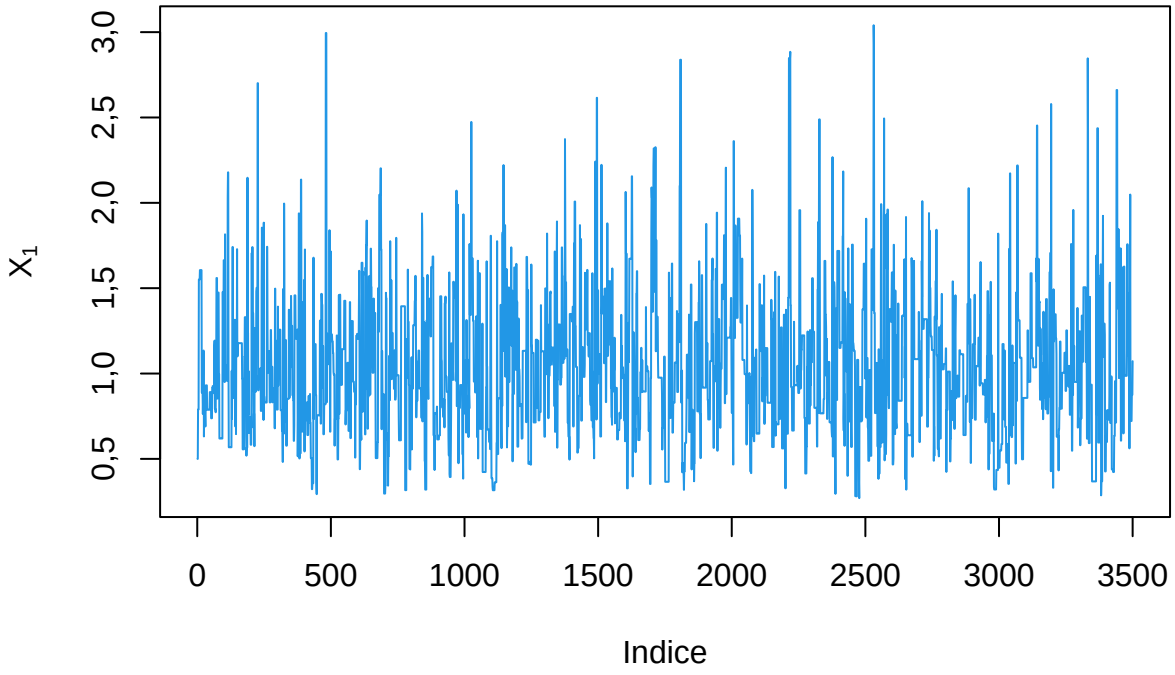
```
##
##  Taxa de aceitação de X1 (%): 35,1
```

```r
cat("\n Taxa de aceitação de X2 (%):", round(nac2 / (nsim - 1) * 100, 1))
```

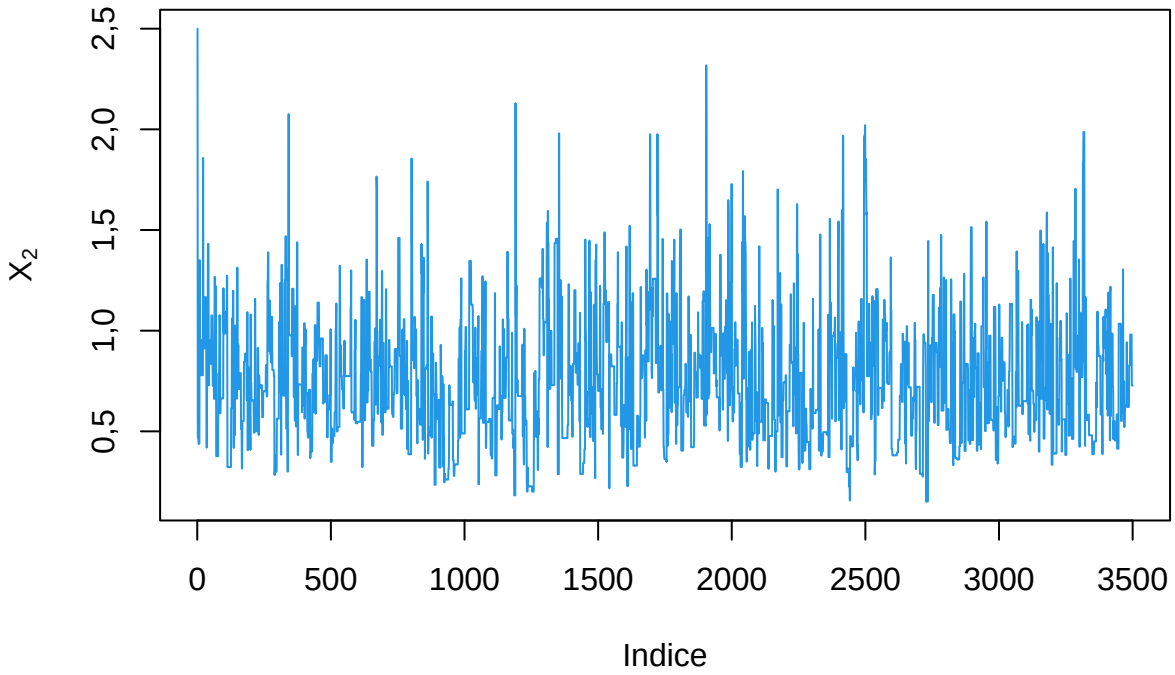
```
##
##  Taxa de aceitação de X2 (%): 35,2
```

Em seguida as sequências geradas são apresentadas graficamente.

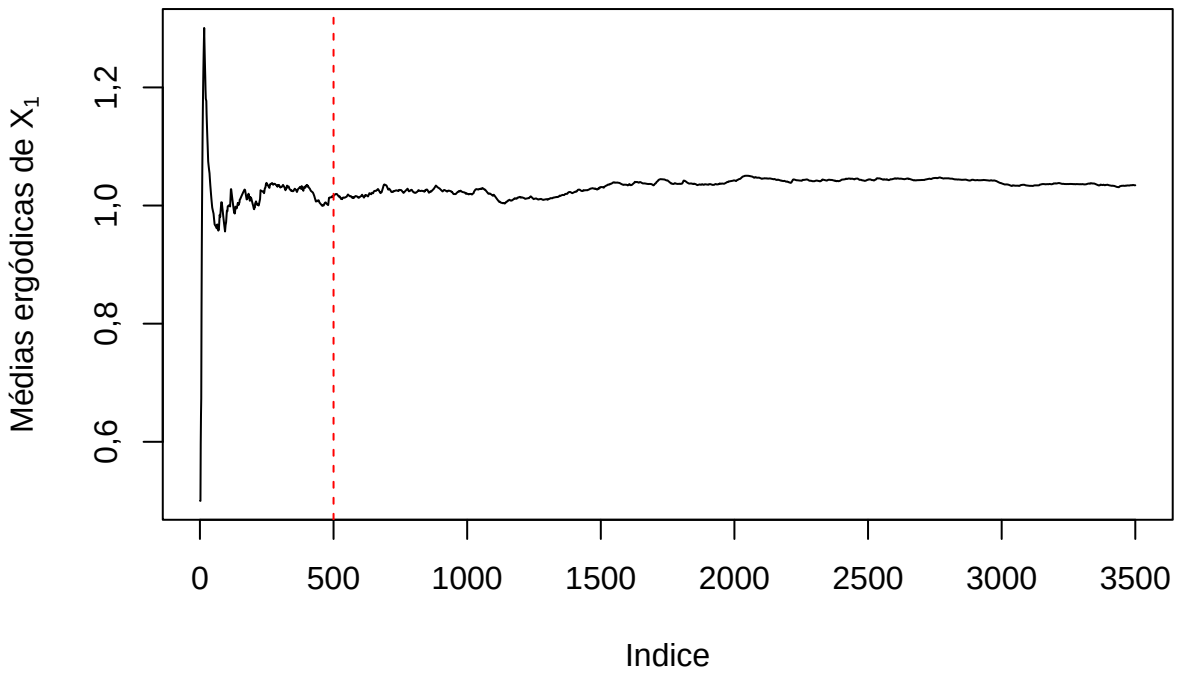
```r
## Gráficos cadeia 1
# Sequência
plot(x11, type = "l", xlab = "Indice", ylab = expression(X[1]), col = 4)
```



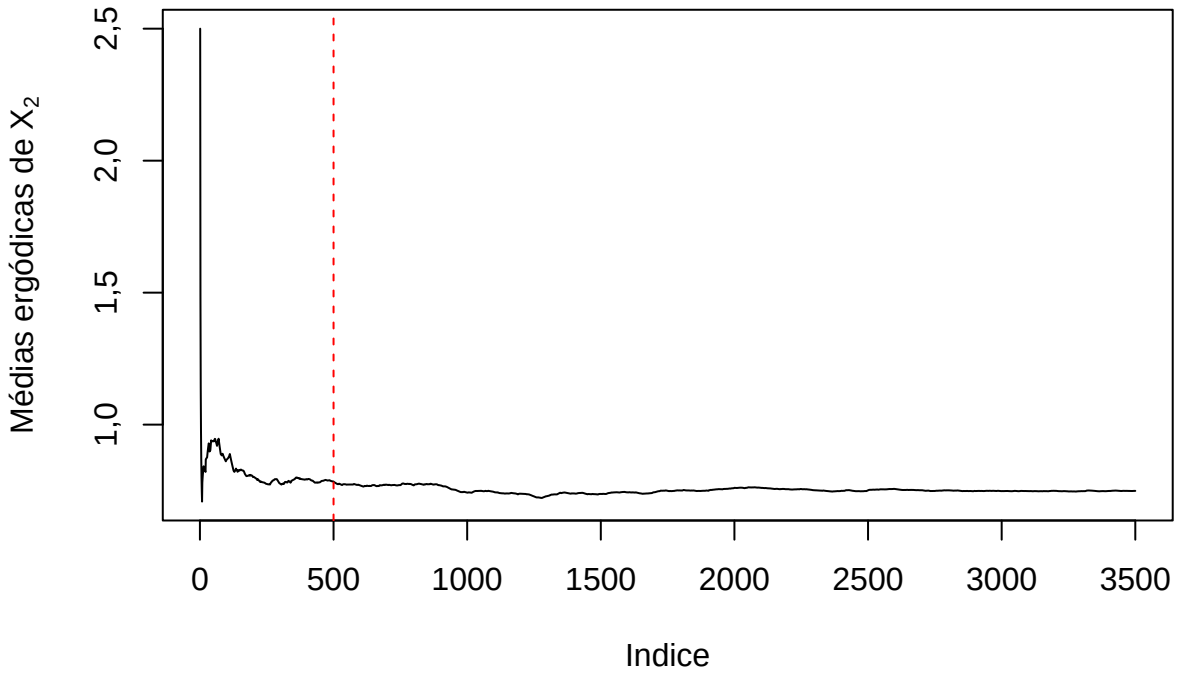

```r
plot(x21, type = "l", xlab = "Indice", ylab = expression(X[2]), col = 4)
```



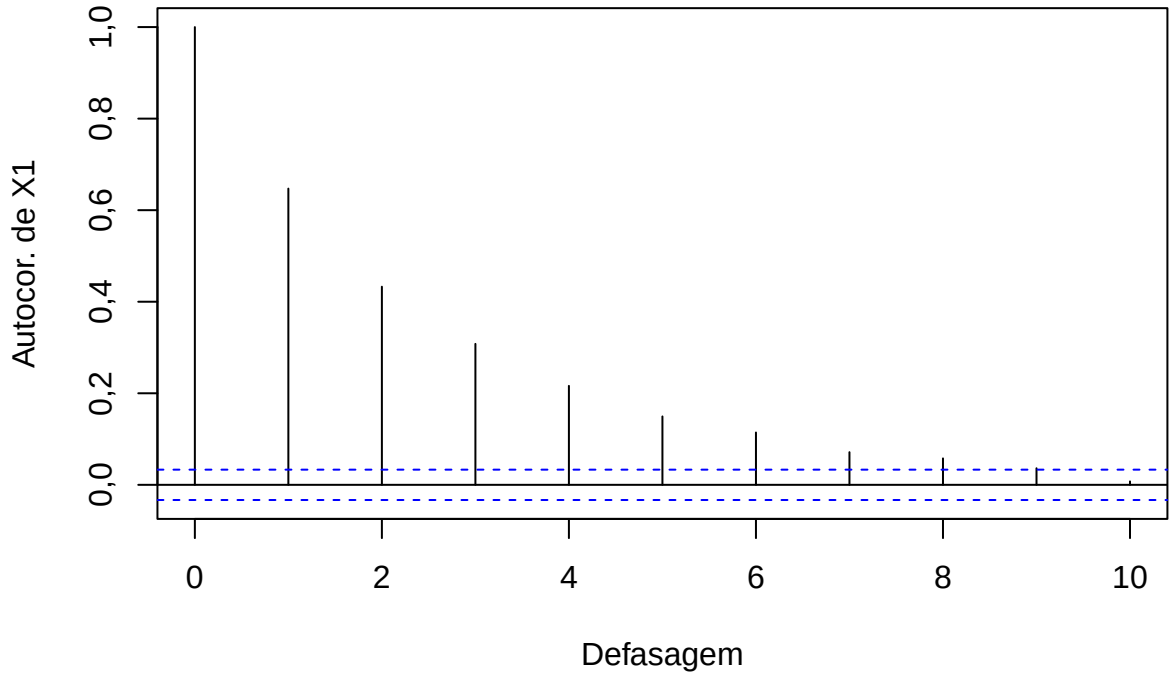

```r
# Médias ergódicas
plot(cumsum(x11) / (1:nsim), type = "l", xlab = "Indice",
     ylab = expression(paste("Médias ergódicas de ", X[1])))
abline(v = descarte, lty = 2, col = "red")
```



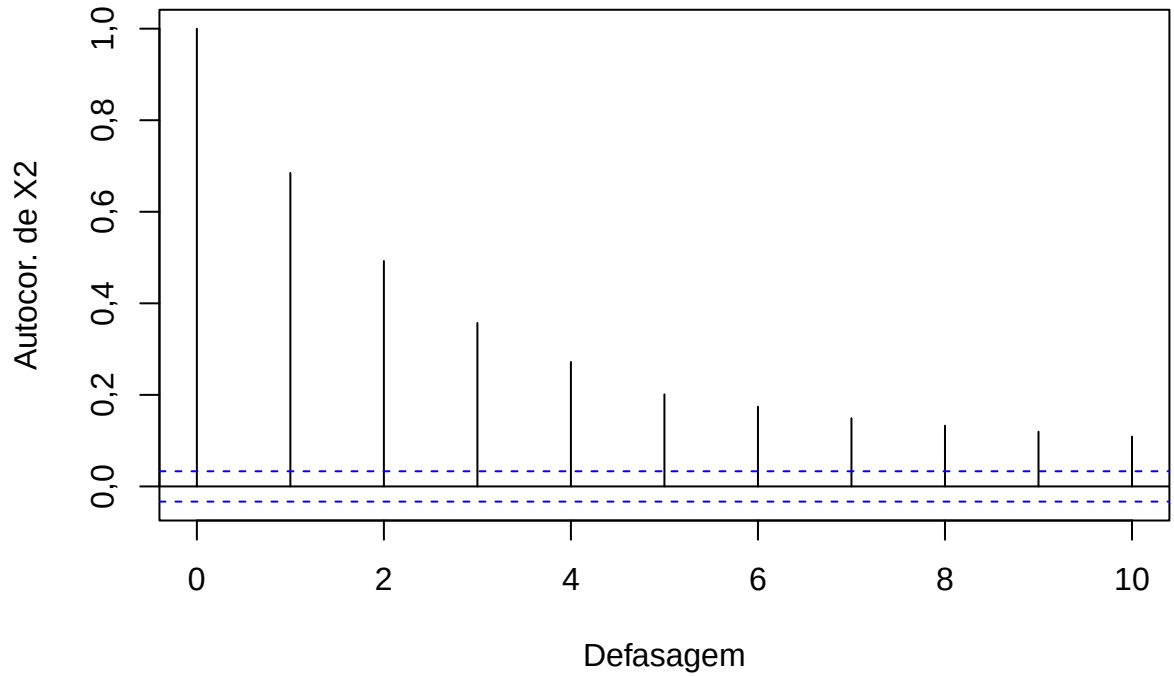

```r
plot(cumsum(x21) / (1:nsim), type = "l", xlab = "Indice",
     ylab = expression(paste("Médias ergódicas de ", X[2])))
abline(v = descarte, lty = 2, col = "red")
```



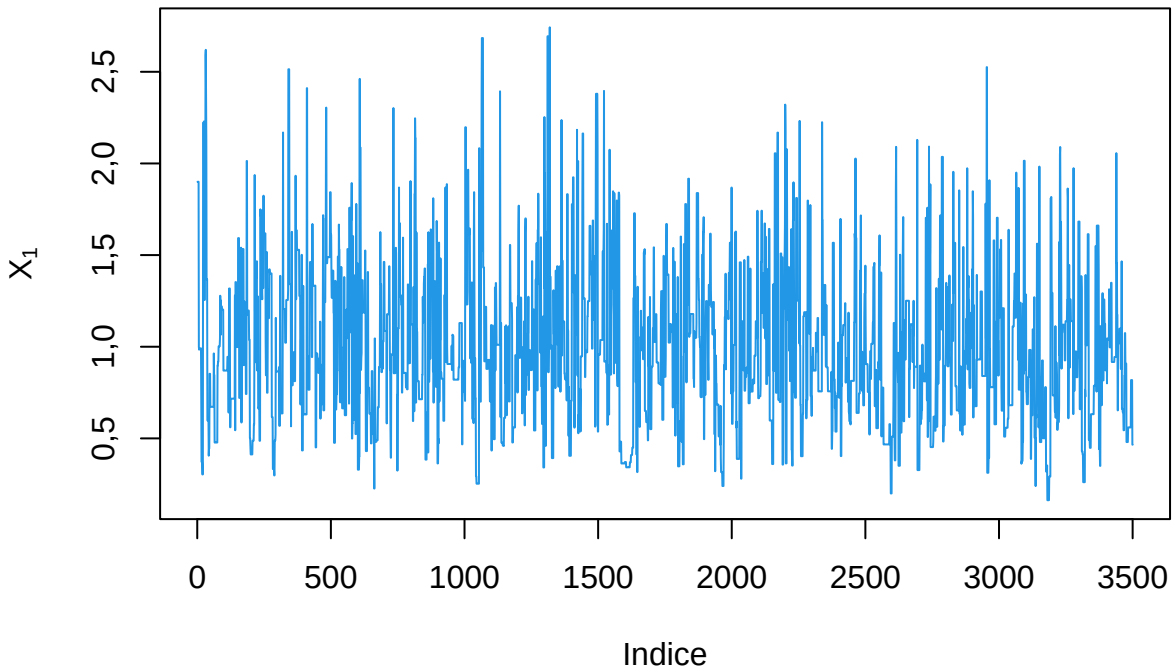

```r
# Autocorrelação
acf(x11, lag.max = 10, main = "", xlab = "Defasagem",
    ylab = "Autocor. de X1")
```



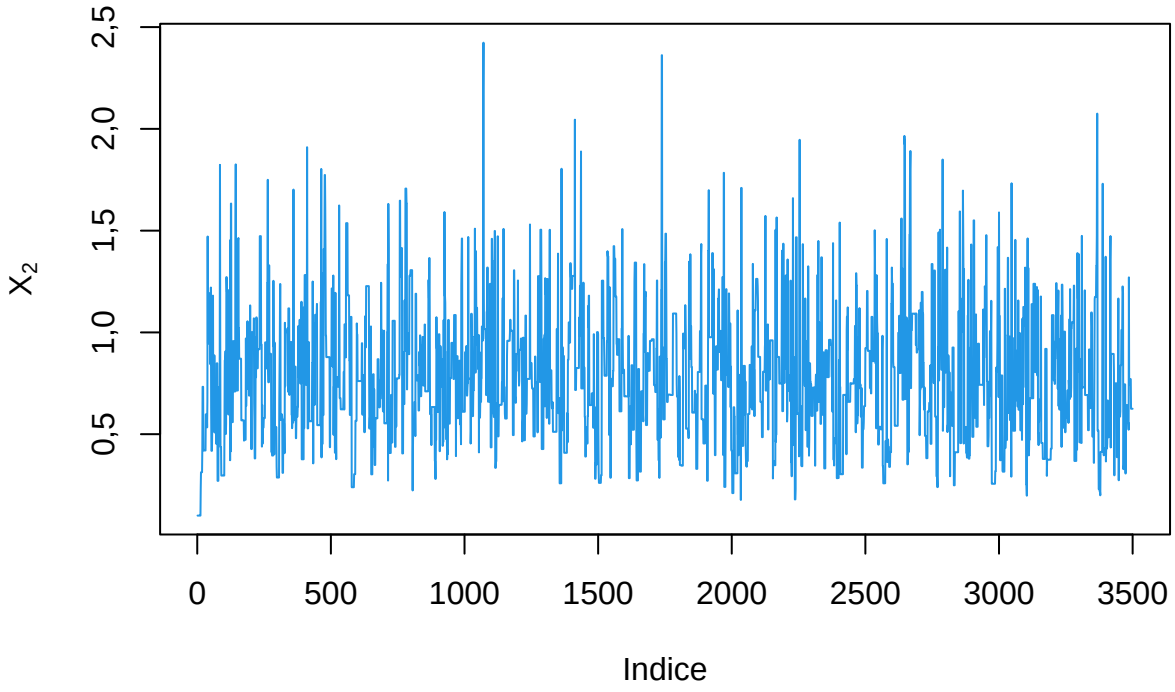

```r
acf(x21, lag.max = 10, main = "", xlab = "Defasagem",
    ylab = "Autocor. de X2")
```



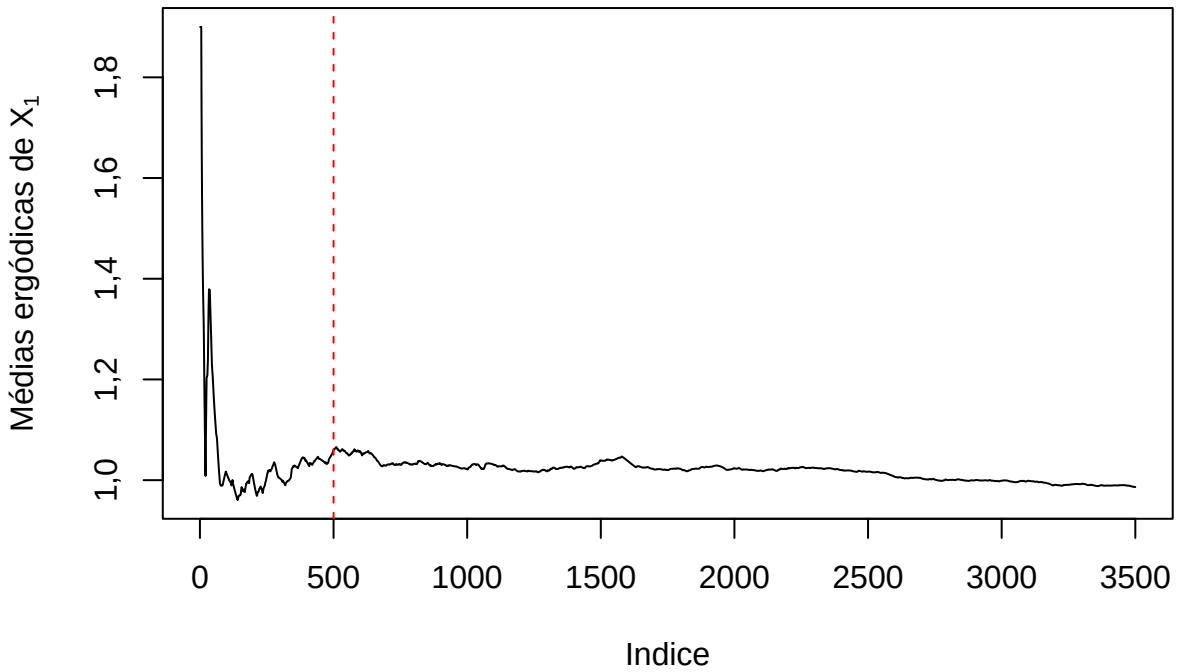

```
## Gráficos cadeia 2
# Sequência
plot(x12, type = "l", xlab = "Indice", ylab = expression(X[1]), col = 4)
```



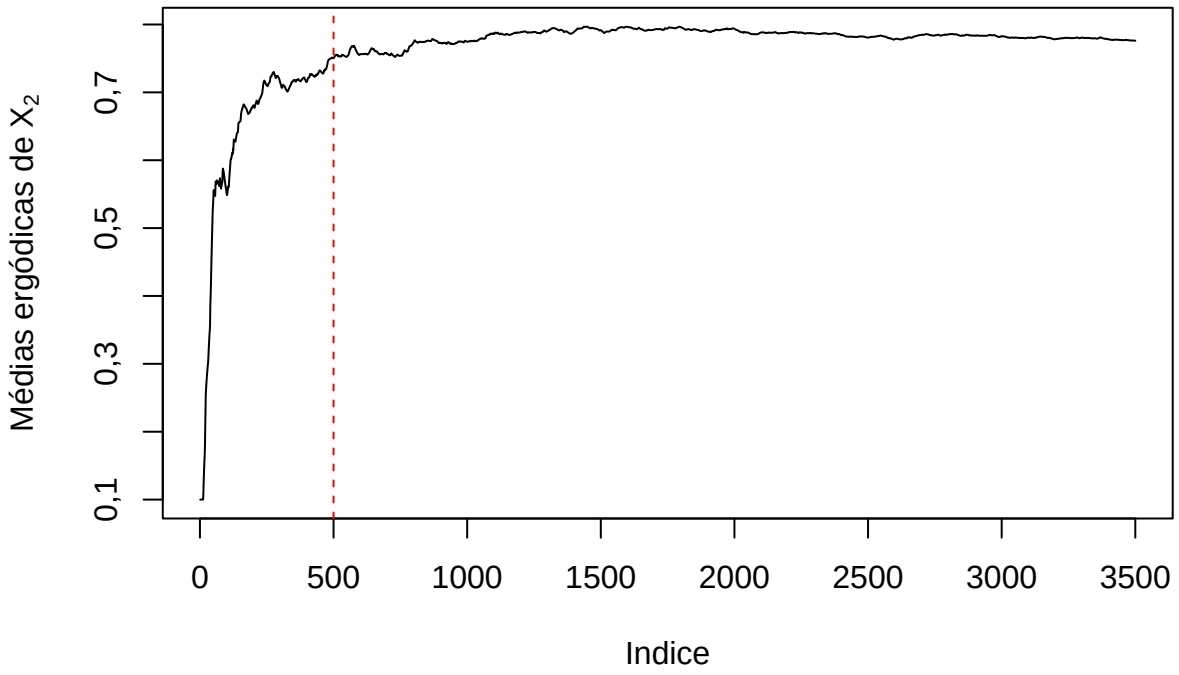

```
plot(x22, type = "l", xlab = "Indice", ylab = expression(X[2]), col = 4)
```



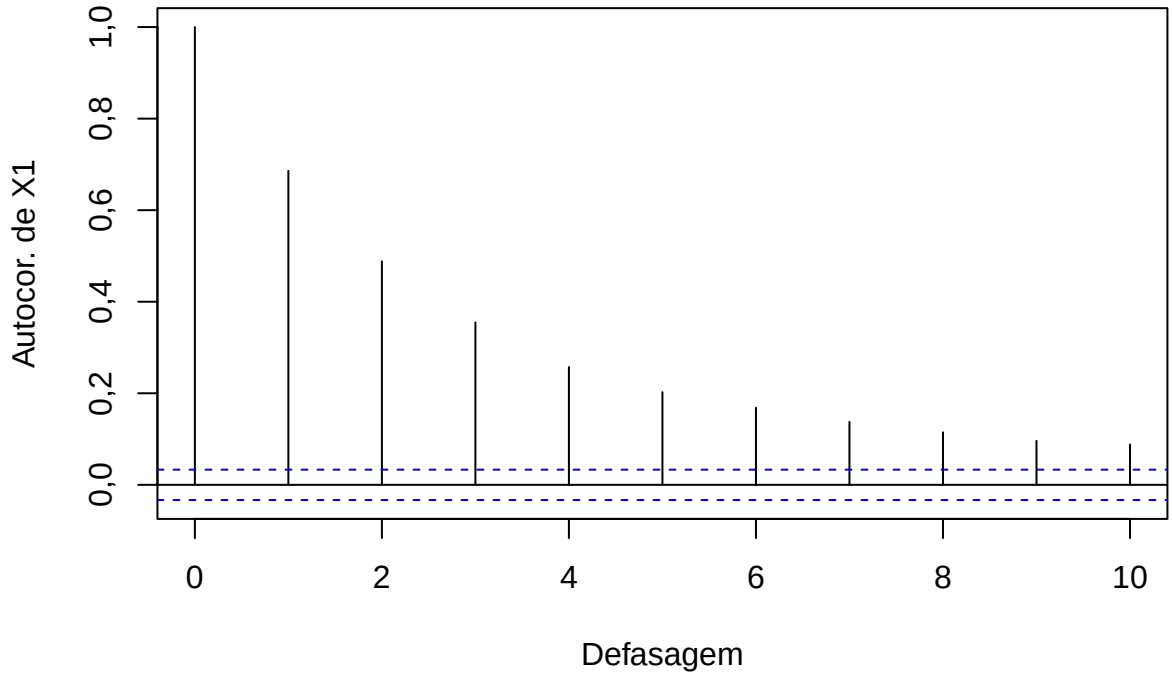

```r
# Médias ergódicas
plot(cumsum(x12) / (1:nsim), type = "l", xlab = "Indice",
     ylab = expression(paste("Médias ergódicas de ", X[1])))
abline(v = descarte, lty = 2, col = "red")
```



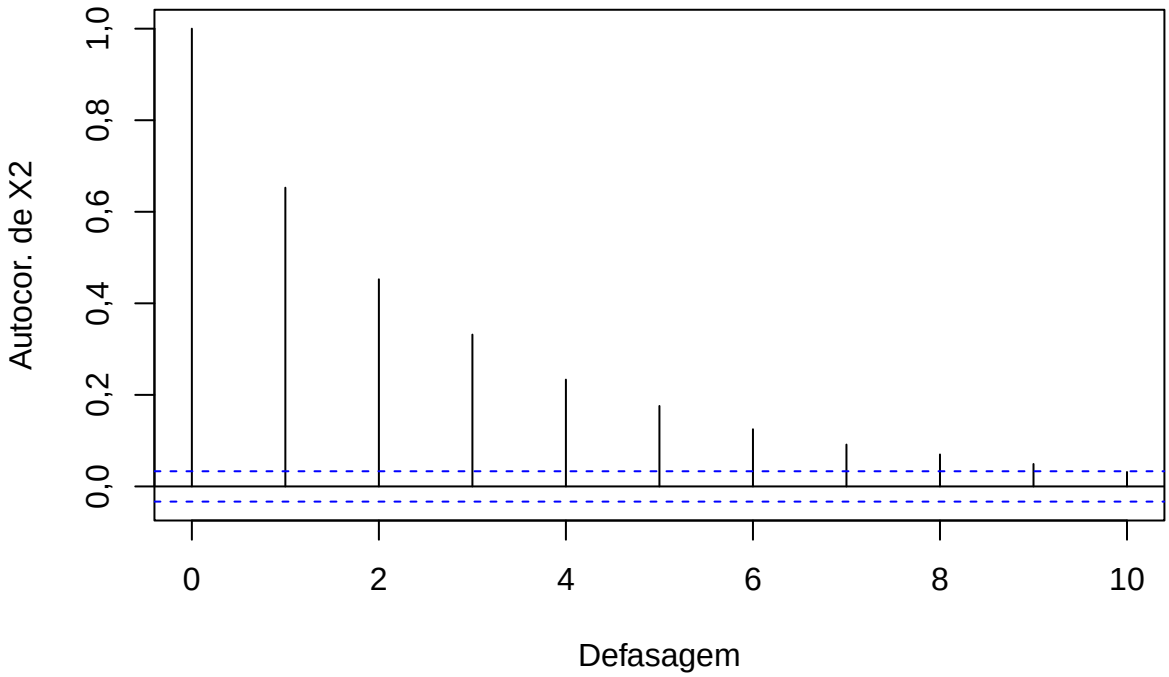

```r
plot(cumsum(x22) / (1:nsim), type = "l", xlab = "Indice",
     ylab = expression(paste("Médias ergódicas de ", X[2])))
abline(v = descarte, lty = 2, col = "red")
```




```
# Autocorrelação
acf(x12, lag.max = 10, main = "", xlab = "Defasagem",
     ylab = "Autocor. de X1")
```



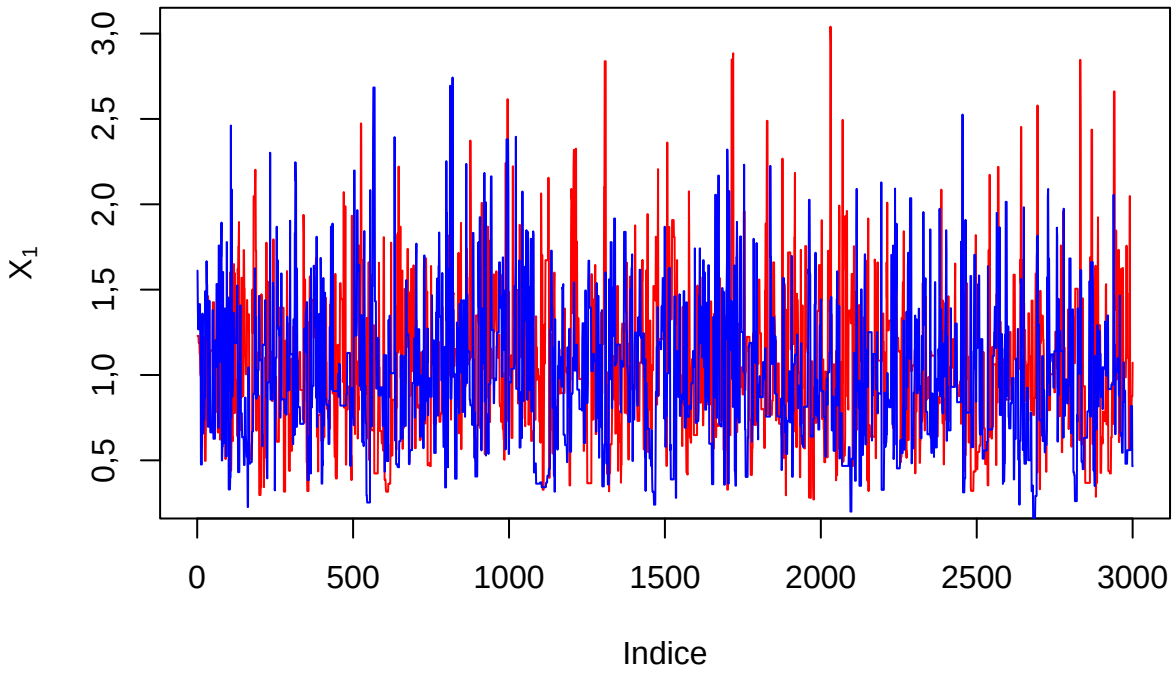

```
acf(x22, lag.max = 10, main = "", xlab = "Defasagem",
     ylab = "Autocor. de X2")
```



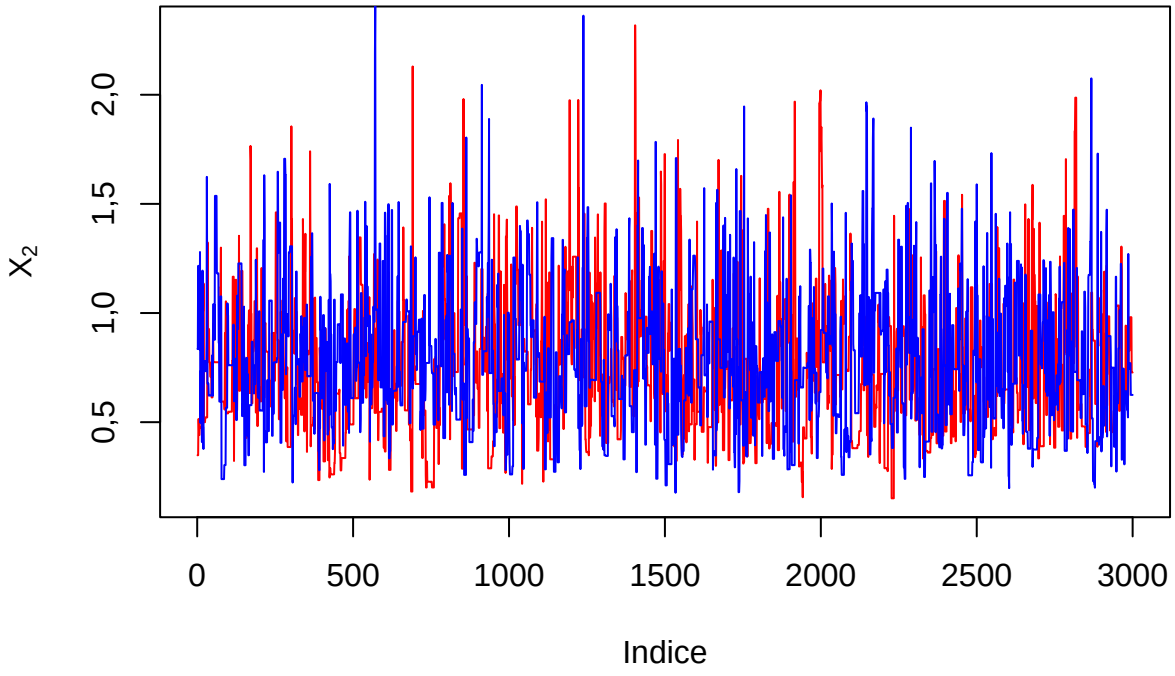
Apresentamos resultados para as sequências após levar em conta descarte (*burn-in*) e espaçamento (*thinning*).

```
# Amostras (descarte e espaçamento levados em conta)
indices <- seq(descarte + 1, nsim, by = espac)
x11 <- x11[indices]
x21 <- x21[indices]
x12 <- x12[indices]
x22 <- x22[indices]
```

```
# Sequência
plot(x11, type = "l", xlab = "Indice", col = "red",
     ylab = expression(X[1]))
lines(x12, col = "blue")
```




```
plot(x21, type = "l", xlab = "Indice", col = "red",
     ylab = expression(X[2]))
lines(x22, col = "blue")
```



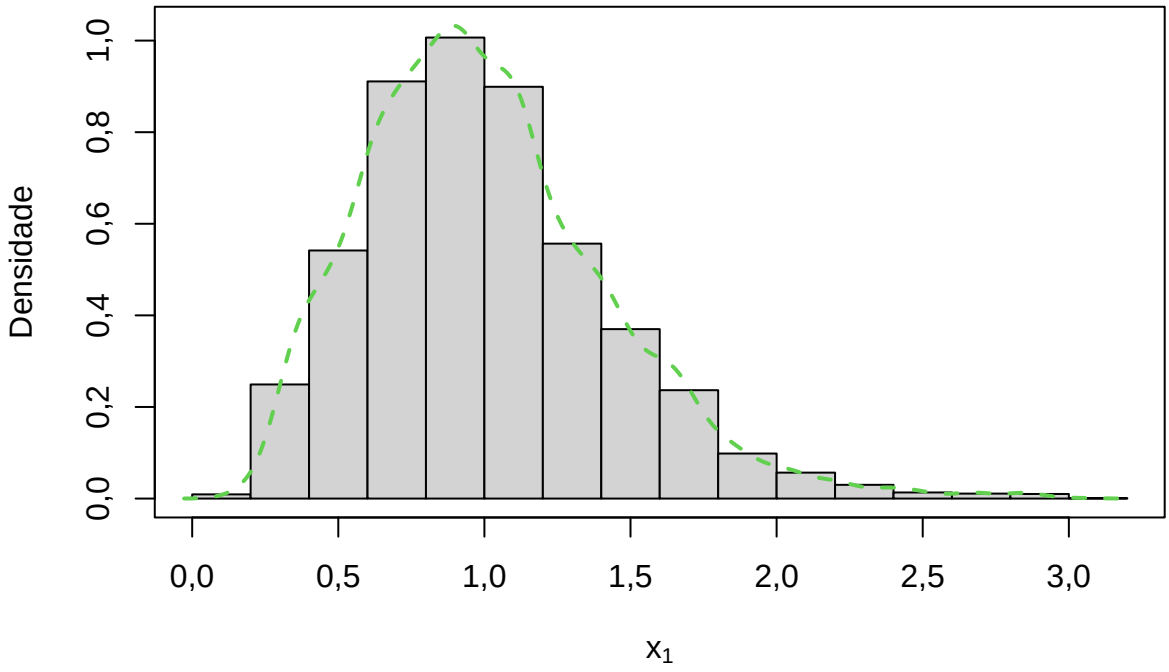
As duas cadeias geradas são combinadas para formar a amostra de tamanho $2M(= 6000)$.

```
# Amostra final
x1 <- c(x11, x12)
x2 <- c(x21, x22)
for (nome in c("x1", "x2")) {
   cat("\n Estatísticas descritivas de", nome, "\n")
   print(summary(get(nome)))
```

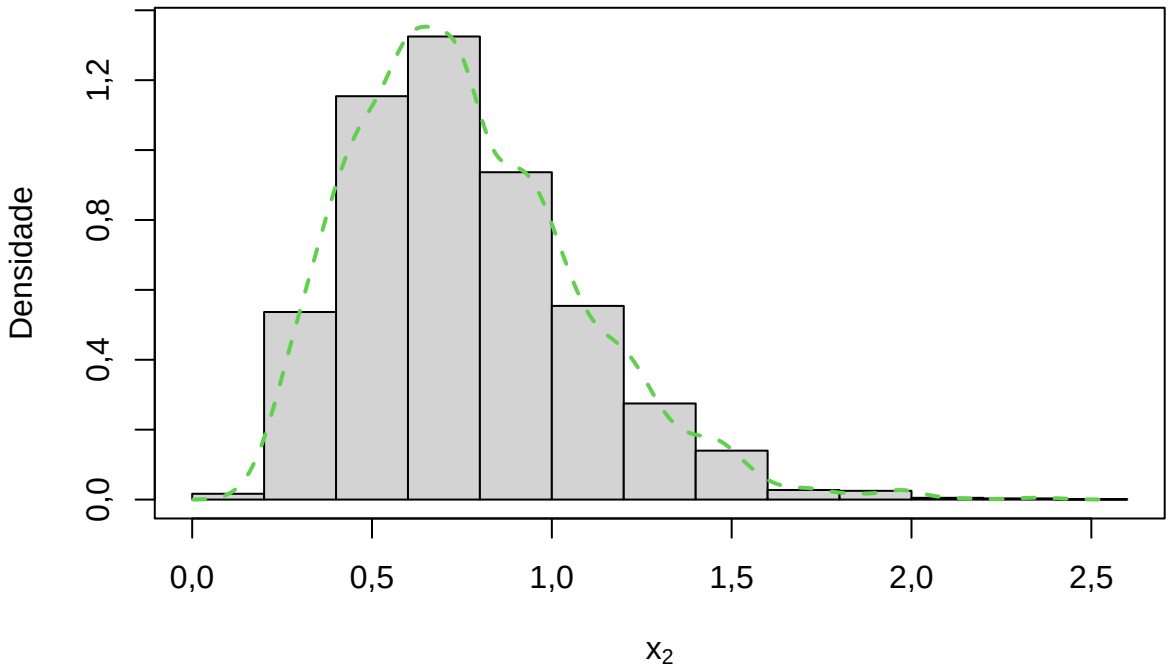
```
}
```

```
##
##  Estatísticas descritivas de x1
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0,1628  0,7070  0,9512  1,0057  1,2452  3,0405
##
##  Estatísticas descritivas de x2
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0,1512  0,5326  0,7185  0,7617  0,9524  2,4235
```

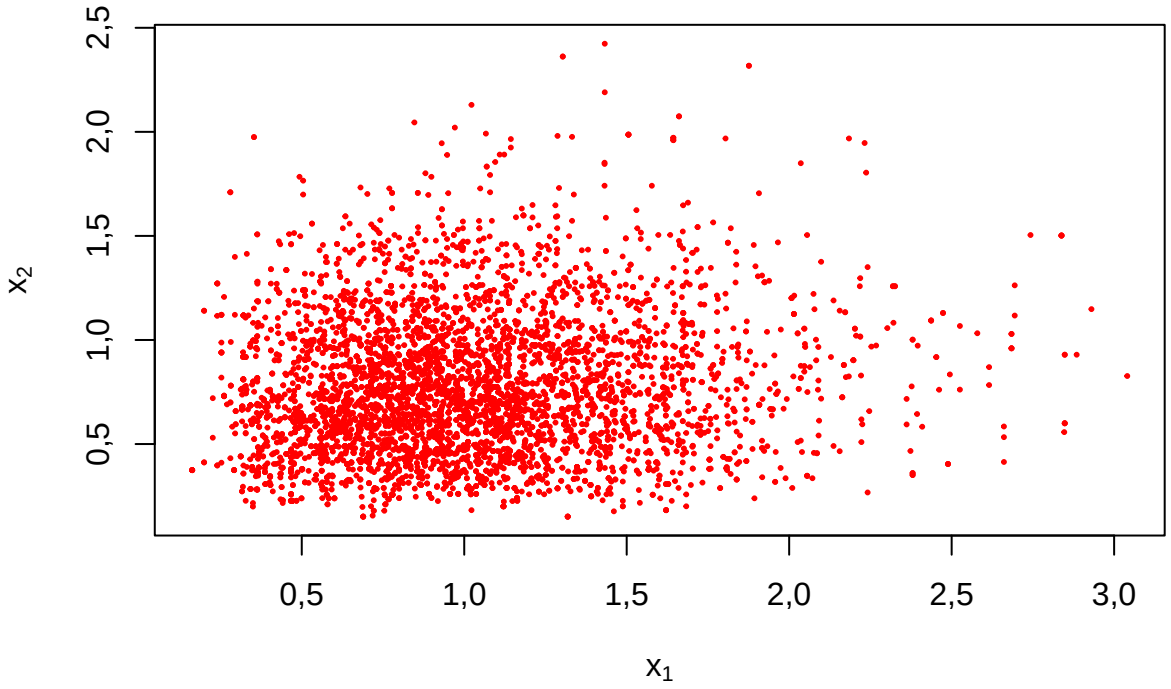
```
# X1
fhist <- hist(x1, plot = FALSE)
fdens <- density(x1)
inty <- c(0, max(fhist$density, fdens$y))
plot(fhist, freq = FALSE, main = "", xlab = expression(x[1]),
     ylab = "Densidade", ylim = inty)
lines(fdens, lty = 2, col = 99, lwd = 2)
box()
```




```
# 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 = 99, lwd =2)
box()
```




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



Nota 1 Para evitar que candidatos `y1` e `y2` com valores muito próximos de 0 sejam gerados, os parâmetros de forma podem ser mudados para `shape = max(1.1, a10 * x11[j - 1])` e `shape = max(1.1, a20 * x21[j - 1])`, respectivamente.

Nota 2 Modifique as constantes para o amostrador (pag. 3) de modo a obter sequências menos autocorrelacionadas e com médias ergódicas mais estáveis em relação às que foram apresentadas.

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

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

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.

Nota 4 Modifique os parâmetros das distribuições propostas (pag. 3) de modo a avaliar a sensibilidade dos resultados.