

Amostra aleatória da distribuição $N(0, 1)$ (em linguagem R)

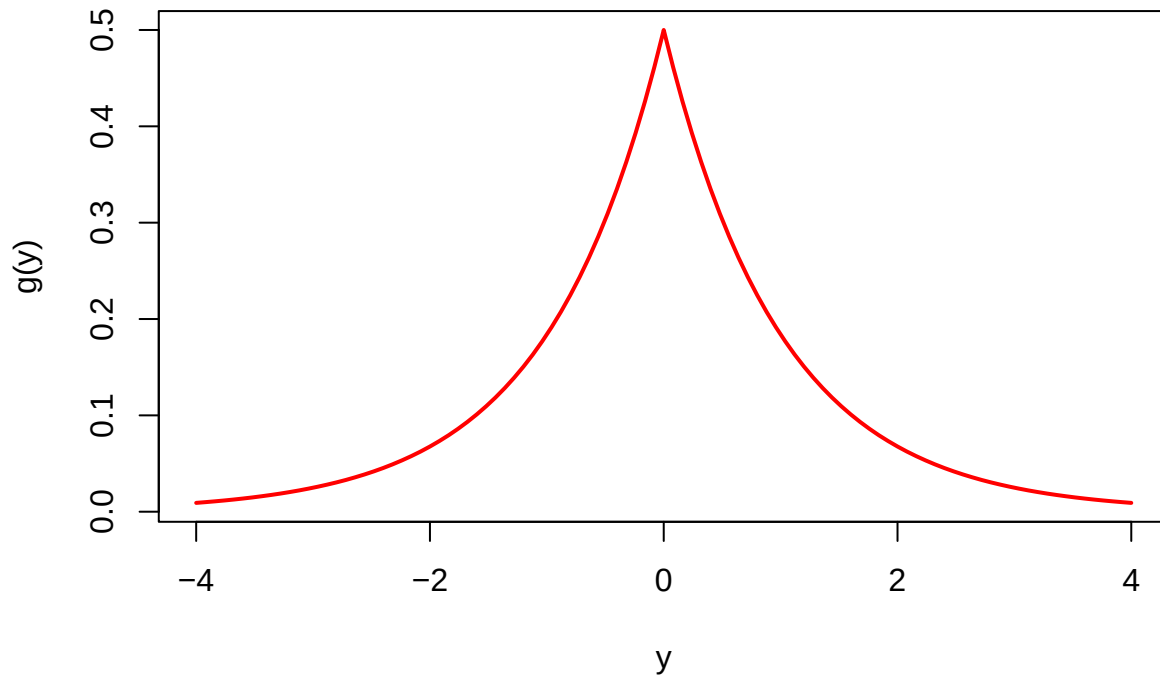
2026

O método de aceitação-rejeição será aplicado na geração de uma amostra pseudoaleatória da distribuição normal padrão, $N(0, 1)$. A variável auxiliar Y tem distribuição Laplace padrão, com função densidade

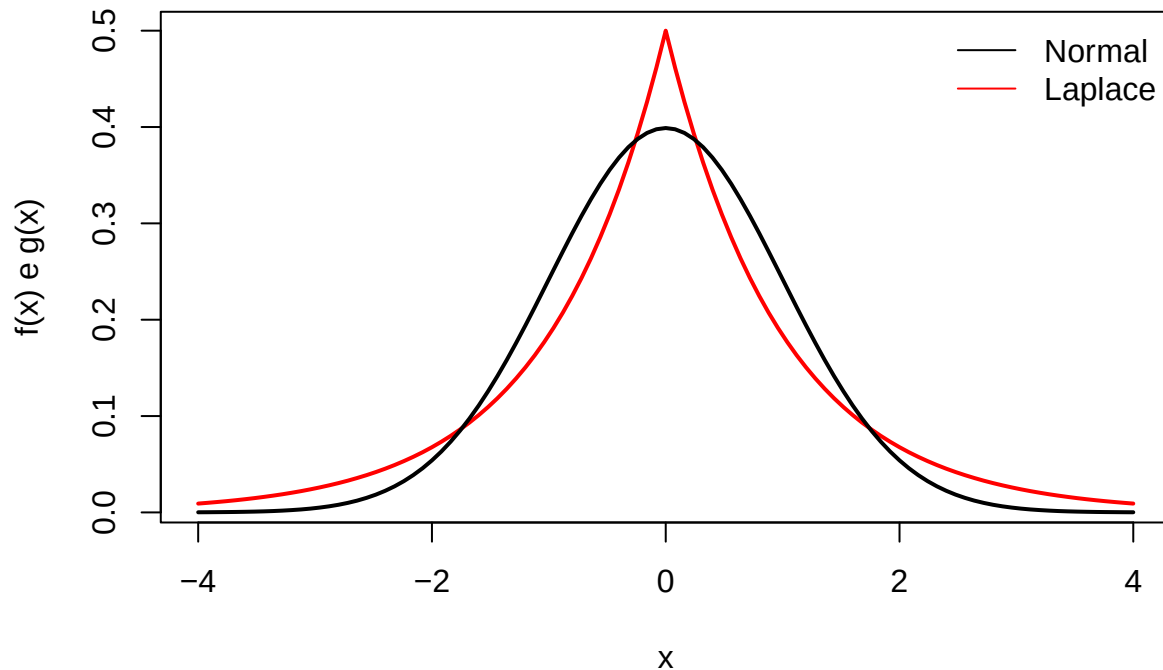
$$g(y) = \frac{1}{2} \exp(-|y|), \quad y \in \mathbb{R}.$$

e gráfico mostrado abaixo.

```
# Função densidade da dist. Laplace padrão
dlap <- function(x) {
  return(0.5 * exp(-abs(x)))
}
curve(dlap, -4, 4, xlab = "y", ylab = "g(y)", col = "red", lwd = 2)
```



```
# Funções densidade
curve(dlap, -4, 4, xlab = "x", ylab = "f(x) e g(x)", col = "red", lwd = 2)
curve(dnorm, add = TRUE, col = "black", lwd = 2)
legend("topright", c("Normal", "Laplace"), col = c("black", "red"),
      lty = 1, bty = "n")
```



A função distribuição de Y e sua inversa são apresentadas abaixo.

$$F_Y(y) = \begin{cases} \frac{1}{2} \exp(y), & \text{se } y \leq 0, \\ 1 - \frac{1}{2} \exp(-y), & \text{se } y > 0, \end{cases} \quad e$$

$$F_Y^{-1}(u) = \begin{cases} \log(2u), & \text{se } 0 < u \leq \frac{1}{2}, \\ -\log(2(1-u)), & \text{se } \frac{1}{2} < u < 1. \end{cases}$$

Sendo assim, geramos amostras de Y pelo método de inversão.

Em seguida é apresentada a função $f(x)/g(x)$ e o cálculo de seu valor máximo M .

```
# Função f(x) / g(x)
fgx <- function(x) {
  return(dnorm(x, 0, 1) / dlap(x))
}
```

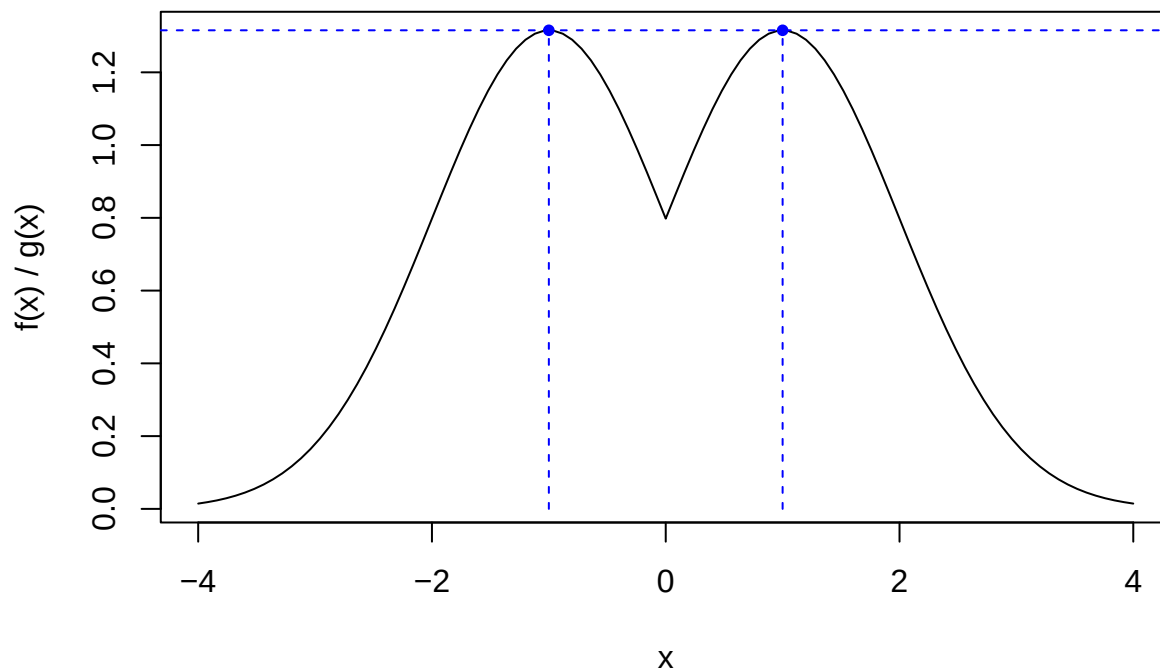
Nota 1. Prove que $M = (2e/\pi)^{1/2}$, para $x \in \{-1, 1\}$.

```
M <- fgx(-1)
cat("\n M =", M)
```

```
##
```

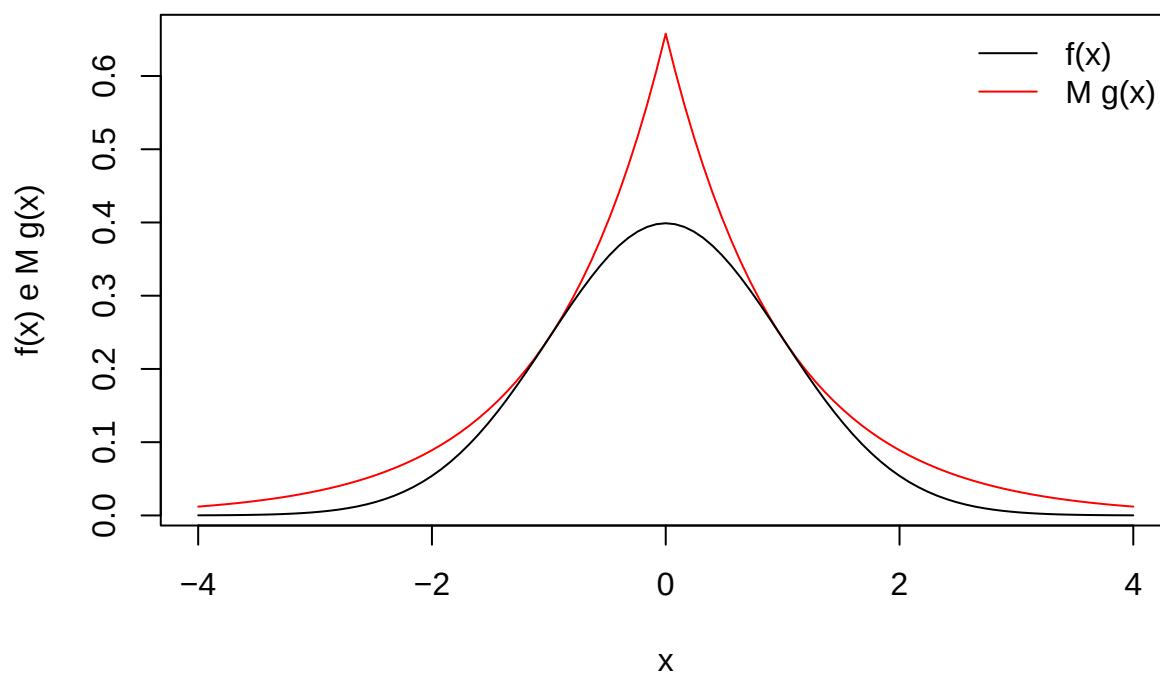
```
## M = 1.315489
```

```
# Gráfico de f(x) / g(x)
curve(fgx, -4, 4, ylab = "f(x) / g(x)")
points(c(-1, 1), c(M, M), pch = 20, col = "blue")
abline(h = M, lty = 2, col = "blue")
segments(c(-1, 1), c(0, 0), c(-1, 1), c(M, M), lty = 2, col = "blue")
```



O gráfico abaixo mostra a função densidade $f(x)$ da variável aleatória X e também $Mg(x)$. Dizemos que $Mg(x)$ é um cobertor (*blanket*) para $f(x)$. De outra forma, $Mg(x)$ envelopa ou domina $f(x)$.

```
# M g(x)
Mg <- function(x, M = M) {
  return(M * dlap(x))
}
curve(Mg(x, M), -4, 4, col = "red", ylab = "f(x) e M g(x)")
curve(dnorm(x, 0, 1), add = TRUE)
legend("topright", c("f(x)", "M g(x)"), col = c("black", "red"),
      lty = 1, bty = "n")
```



```

## Geração da a.a.
# Número de observações
n <- 100

# Geração de uma a.a. de X
set.seed(61) # semente
nger <- n0 <- 0
ax <- c()
while (n0 < n) {
  rej <- TRUE
  while(rej) {
    nger <- nger + 1

    # Variável auxiliar (método de inversão)
    u <- runif(1)
    y <- ifelse(u <= 0.5, log(2 * u), -log(2 * (1 - u)))

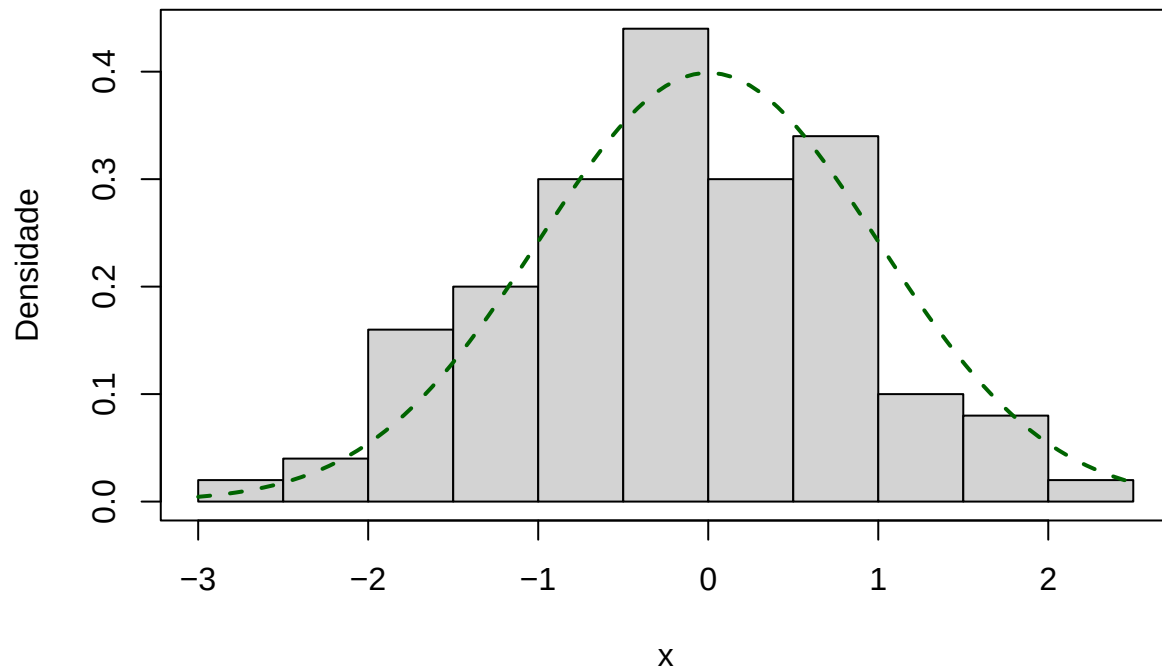
    # Aceitação ou rejeição
    if (runif(1) <= fgx(y) / M) {
      n0 <- n0 + 1
      ax[n0] <- y
      rej <- FALSE
    }
  }
}

cat("\n Tamanho da amostra:", n, "\n Número de tentativas:", nger)

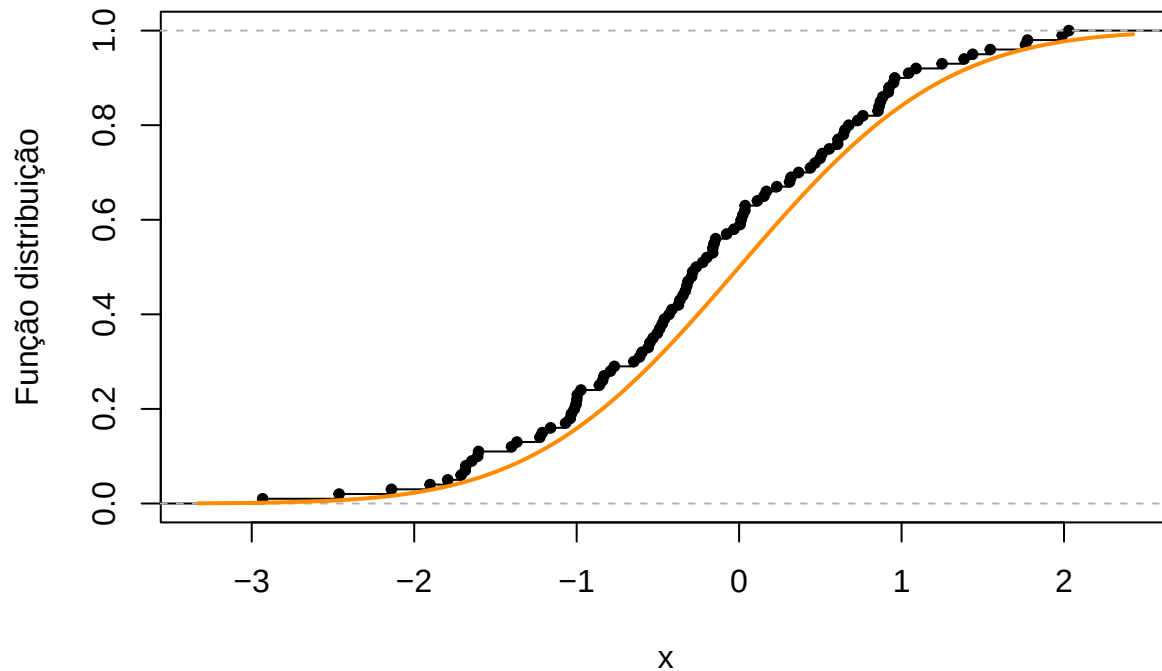
##
## Tamanho da amostra: 100
## Número de tentativas: 131

# Gráficos
hist(ax, freq = FALSE, main = "", xlab = "x", ylab = "Densidade")
curve(dnorm(x, 0, 1), add = TRUE, lty = 2, col = "darkgreen", lwd = 2)
box()

```



```
plot(ecdf(ax), main = "", xlab = "x", ylab = "Função distribuição",
     pch = 20)
curve(pnorm(x, 0, 1), add = TRUE, col = "darkorange", lwd = 2)
```



Nota 2. Os gráficos acima sugerem que a amostra foi gerada de uma distribuição $N(0, 1)$?

```
# Teste de bondade do ajuste
ks.test(ax, "pnorm", 0, 1)
```

```
##
## Asymptotic one-sample Kolmogorov-Smirnov test
##
```

```
## data:  ax
## D = 0.11741, p-value = 0.1269
## alternative hypothesis: two-sided
```

Nota 3. O resultado do teste acima sugere que a amostra foi gerada de uma distribuição $N(0, 1)$?

Apresente outros testes que poderiam ser utilizados para responder à pergunta acima.

Nota 4. Reescreva o código da pag. 3 substituindo **while** por **repeat**.

Nota 5. Refaça o exemplo utilizando a variável aleatória Y com distribuição Cauchy padrão, cuja função densidade é

$$g(y) = \frac{1}{\pi(1 + y^2)}, \quad y \in \mathbb{R}.$$

Nota 6. Considere a distribuição Beta(α, β) em que $\alpha > 1$ e $\beta > 1$. Desenvolva um gerador de amostras para esta distribuição usando o método de aceitação-rejeição.

Nota 7. Refaça os exemplos em linguagem Python.