Distribuições discretas

2024

Algumas distribuições discretas são apresentadas. A linguagem R é utilizada.

Várias distribuições de probabilidade discretas estão implementadas no pacote `stats`, que é um dos pacotes básicos da linguagem R. Em geral, se X denota uma variável aleatória, para uma distribuição qualquer identificada por `nome`, existem quatro funções, brevemente descritas abaixo.

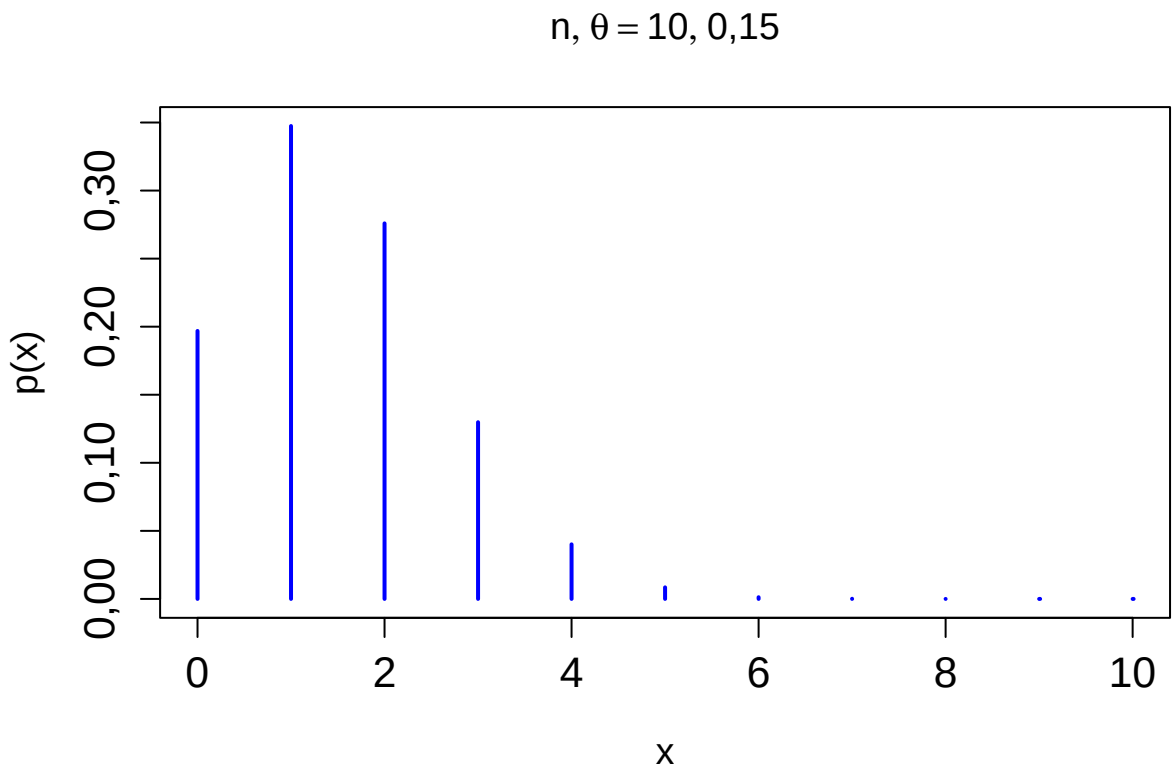
1. `dnome`: Se a distribuição for contínua, significa a função densidade de probabilidade $f(x)$; se a distribuição for discreta, significa a função massa de probabilidade $p(x) = P(X = x)$.
2. `pnome`: função distribuição acumulada $F(x) = P(X \leq x)$.
3. `qnome`: função quantil.
4. `rnome`: geração de amostras da distribuição.

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

1. Binomial

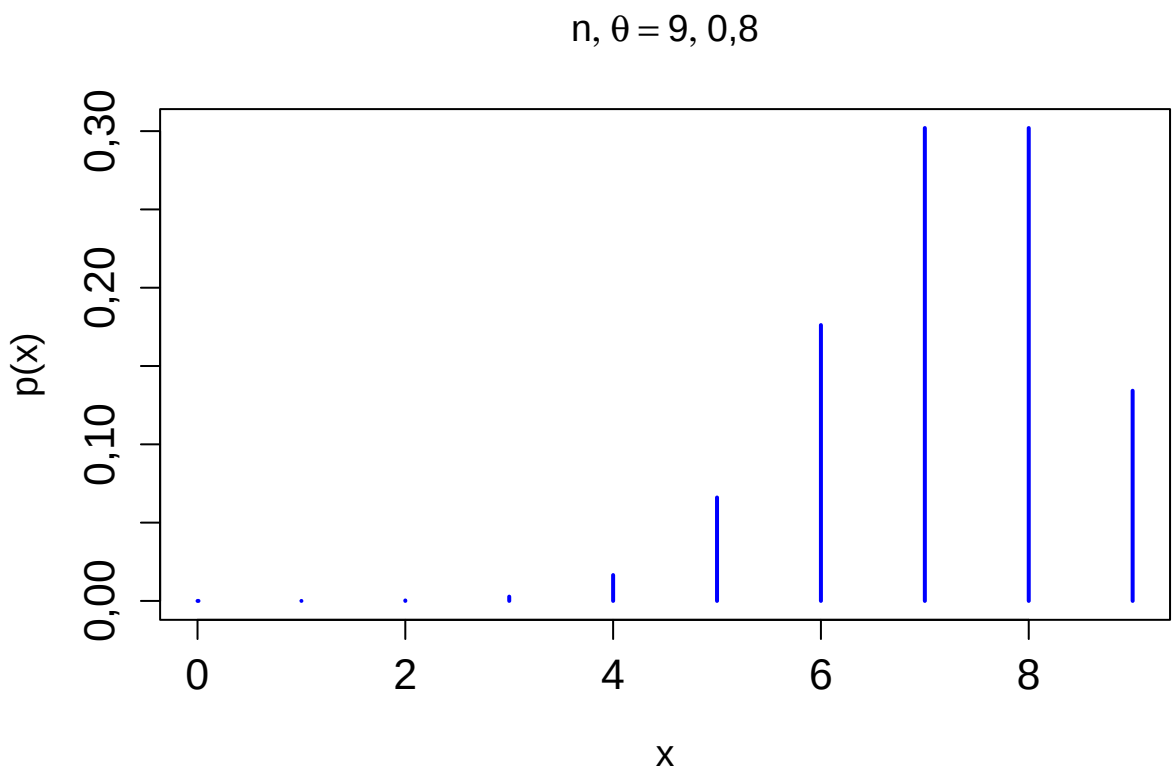
Sintaxe básica: `dbinom(x, n, teta)`.

```
n <- 10
teta <- 0.15
curve(dbinom(x, n, teta), from = 0, to = n, n = n + 1, type = "h",
      xlab = "x", ylab = "p(x)", col = "blue", lwd = 2,
      cex.main = 1.2, cex.axis = 1.3, cex.lab = 1.2,
      main = substitute(list(n, theta) == list(v1, v2),
             list(v1 = n, v2 = teta)))
```

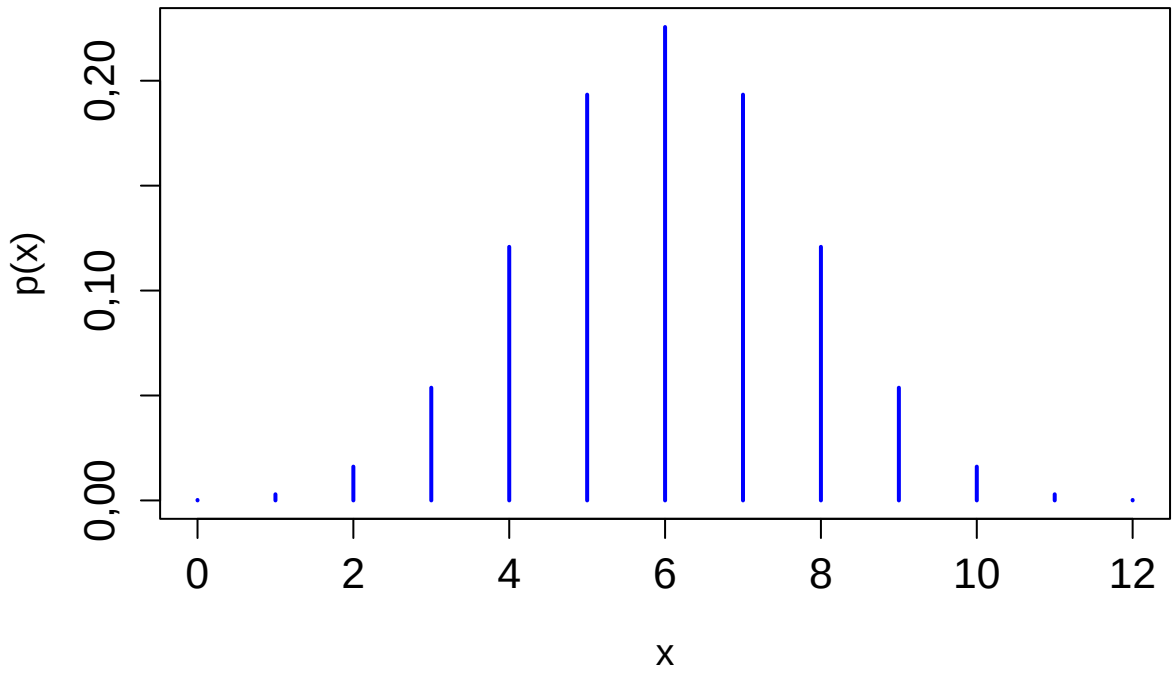


Nota 1. Procure entender todos os argumentos do comando acima.

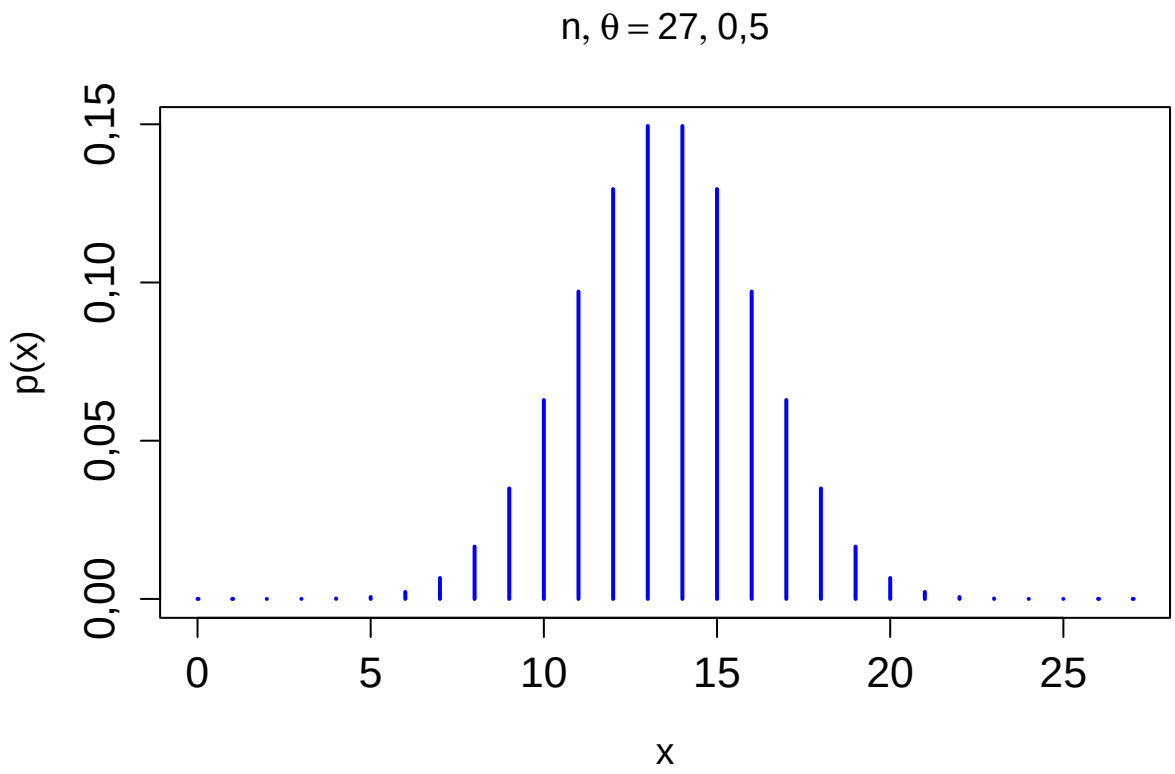
```
n <- 9
teta <- 0.8
curve(dbinom(x, n, teta), from = 0, to = n, n = n + 1, type = "h",
      xlab = "x", ylab = "p(x)", col = "blue", lwd = 2,
      cex.main = 1.2, cex.axis = 1.3, cex.lab = 1.2,
      main = substitute(list(n, theta) == list(v1, v2),
             list(v1 = n, v2 = teta)))
```



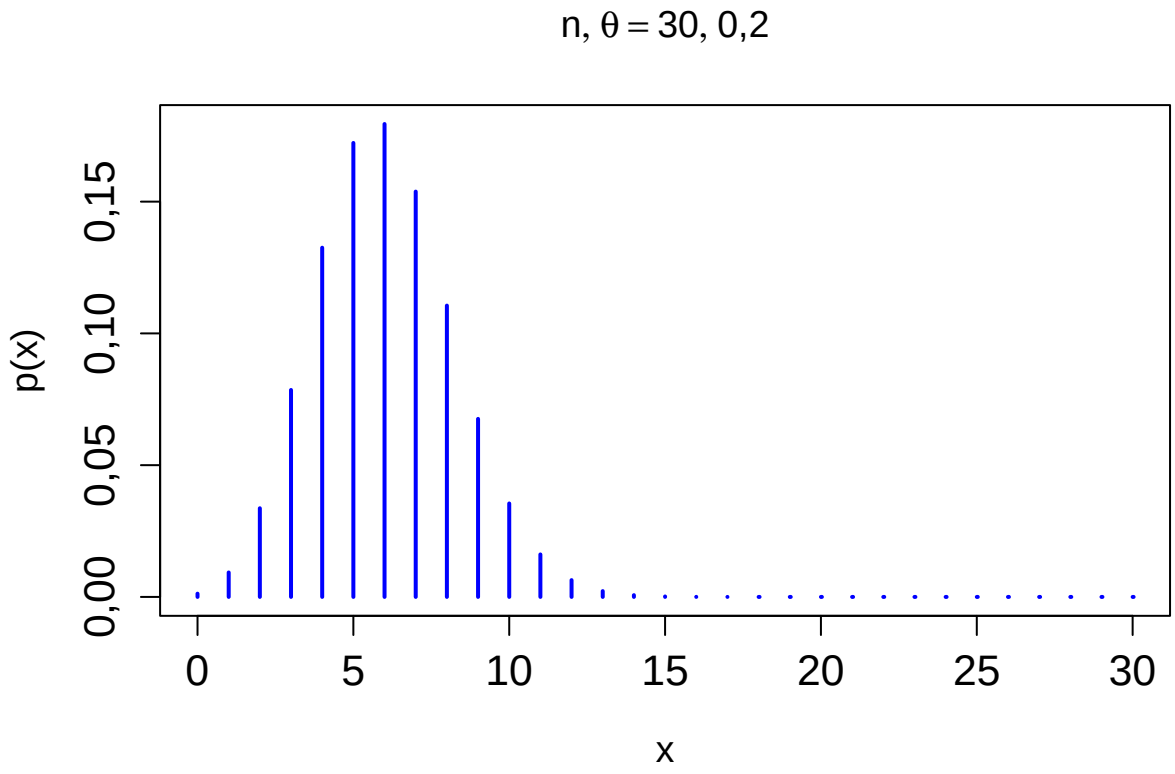

```r
n <- 12
teta <- 0.5
curve(dbinom(x, n, teta), from = 0, to = n, n = n + 1, type = "h",
      xlab = "x", ylab = "p(x)", col = "blue", lwd = 2,
      cex.main = 1.2, cex.axis = 1.3, cex.lab = 1.2,
      main = substitute(list(n, theta) == list(v1, v2),
             list(v1 = n, v2 = teta)))
```




```r
n <- 27
teta <- 0.5
curve(dbinom(x, n, teta), from = 0, to = n, n = n + 1, type = "h",
      xlab = "x", ylab = "p(x)", col = "blue", lwd = 2,
      cex.main = 1.2, cex.axis = 1.3, cex.lab = 1.2,
      main = substitute(list(n, theta) == list(v1, v2),
             list(v1 = n, v2 = teta)))
```




```r
# Prova de múltipla escolha com 30 questões de cinco alternativas.
n <- 30
teta <- 1 / 5
curve(dbinom(x, n, teta), from = 0, to = n, n = n + 1, type = "h",
      xlab = "x", ylab = "p(x)", col = "blue", lwd = 2,
      cex.main = 1.2, cex.axis = 1.3, cex.lab = 1.2,
      main = substitute(list(n, theta) == list(v1, v2),
             list(v1 = n, v2 = teta)))
```



2. Geométrica

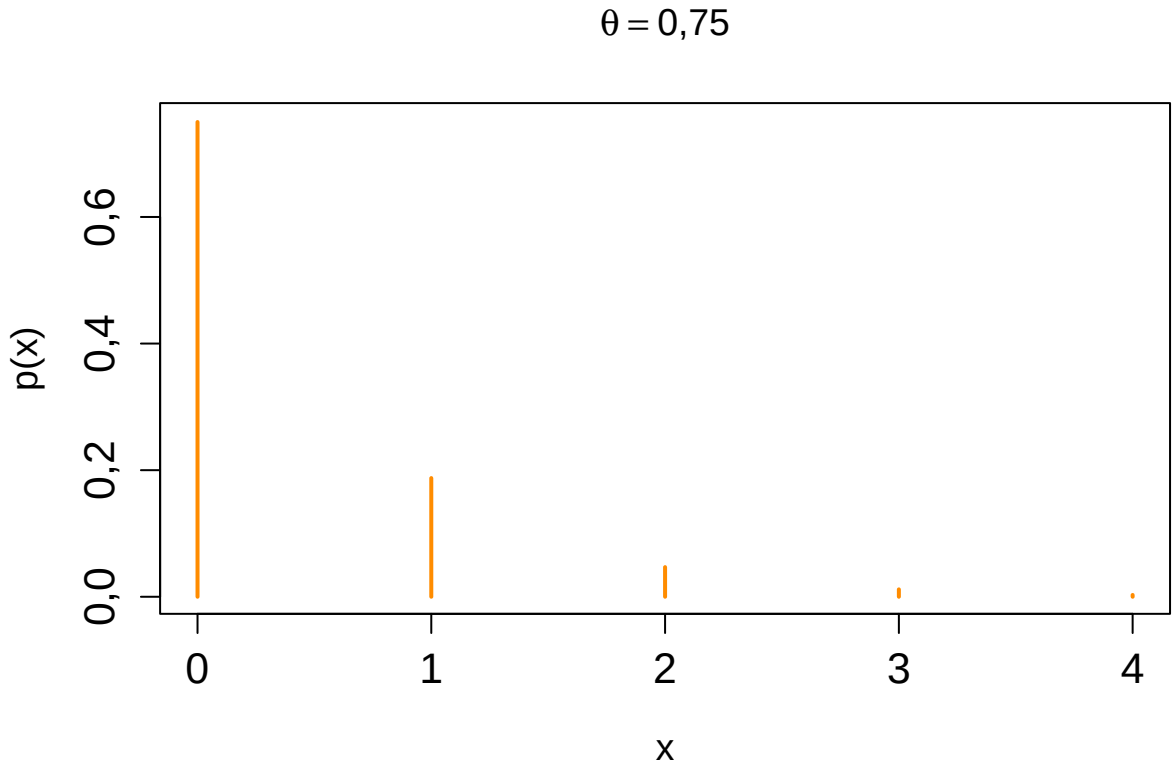
A variável aleatória X se refere ao número de falhas que antecedem o primeiro sucesso, de modo que $x \in \{0, 1, 2, \dots\}$. Nas aulas utilizamos $X + 1$.

Sintaxe básica: `dgeom(x, teta)`.

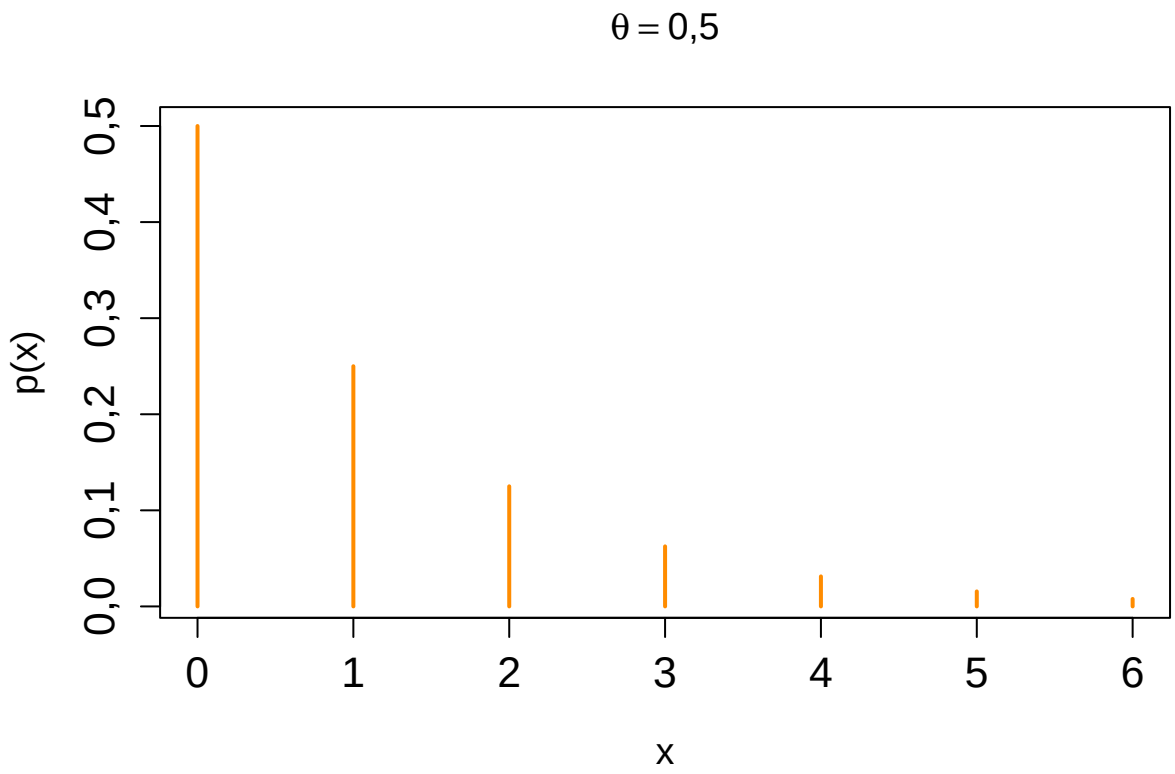
```
teta <- 0.75
xmax <- qgeom(0.999, teta)
```

Na linha acima, `xmax` representa o valor máximo da variável no gráfico e corresponde à probabilidade acumulada de 0,999 (quantil 0,999 da distribuição).

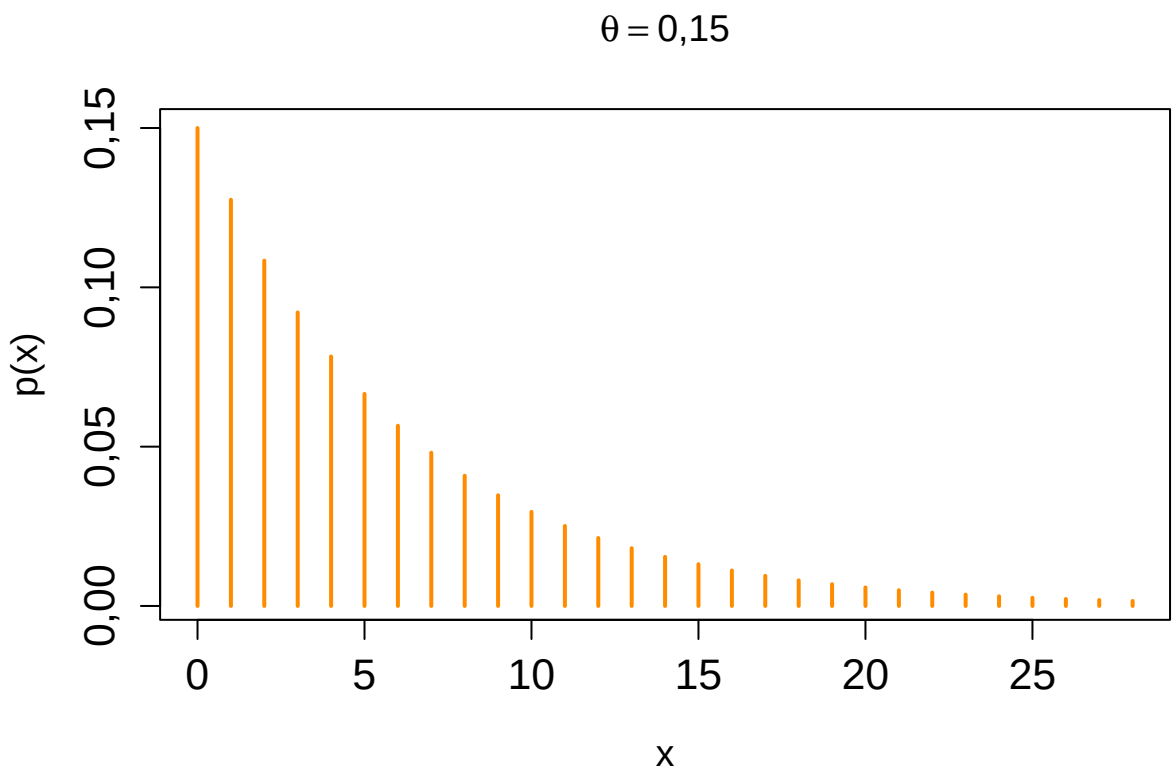
```
curve(dgeom(x, teta), from = 0, to = xmax, n = xmax + 1, type = "h",
    lwd = 2, xlab = "x" , ylab = "p(x)", col = "darkorange",
    main = substitute(list(theta) == list(v1),     list(v1 = teta)),
    cex.main = 1.2, cex.axis = 1.3, cex.lab = 1.2)
```



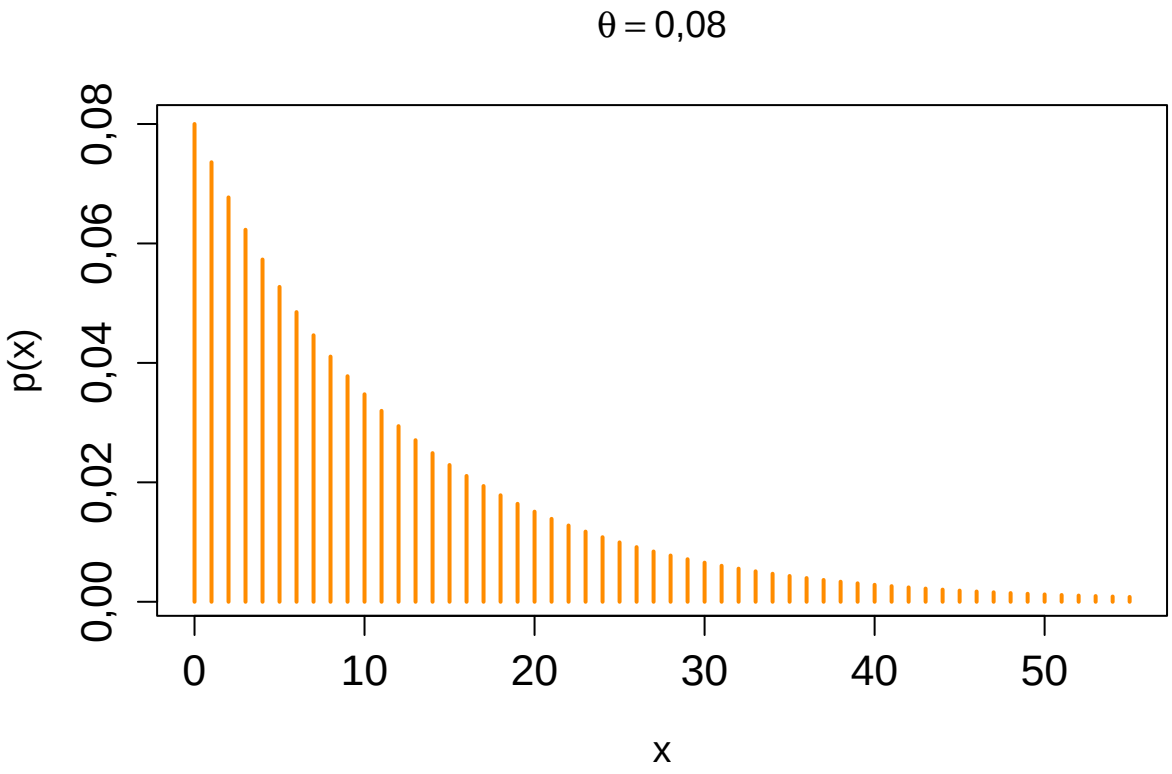

```
teta <- 0.5
xmax <- qgeom(0.99, teta)
curve(dgeom(x, teta), from = 0, to = xmax, n = xmax + 1, type = "h",
    lwd = 2, xlab = "x" , ylab = "p(x)", col = "darkorange",
    main = substitute(list(theta) == list(v1), list(v1 = teta)),
    cex.main = 1.2, cex.axis = 1.3, cex.lab = 1.2)
```




```
teta <- 0.15
xmax <- qgeom(0.99, teta)
curve(dgeom(x, teta), from = 0, to = xmax, n = xmax + 1, type = "h",
    lwd = 2, xlab = "x" , ylab = "p(x)", col = "darkorange",
    main = substitute(list(theta) == list(v1), list(v1 = teta)),
    cex.main = 1.2, cex.axis = 1.3, cex.lab = 1.2)
```




```r
teta <- 0.08
xmax <- qgeom(0.99, teta)
curve(dgeom(x, teta), from = 0, to = xmax, n = xmax + 1, type = "h",
    lwd = 2, xlab = "x" , ylab = "p(x)", col = "darkorange",
    main = substitute(list(theta) == list(v1), list(v1 = teta)),
    cex.main = 1.2, cex.axis = 1.3, cex.lab = 1.2)
```

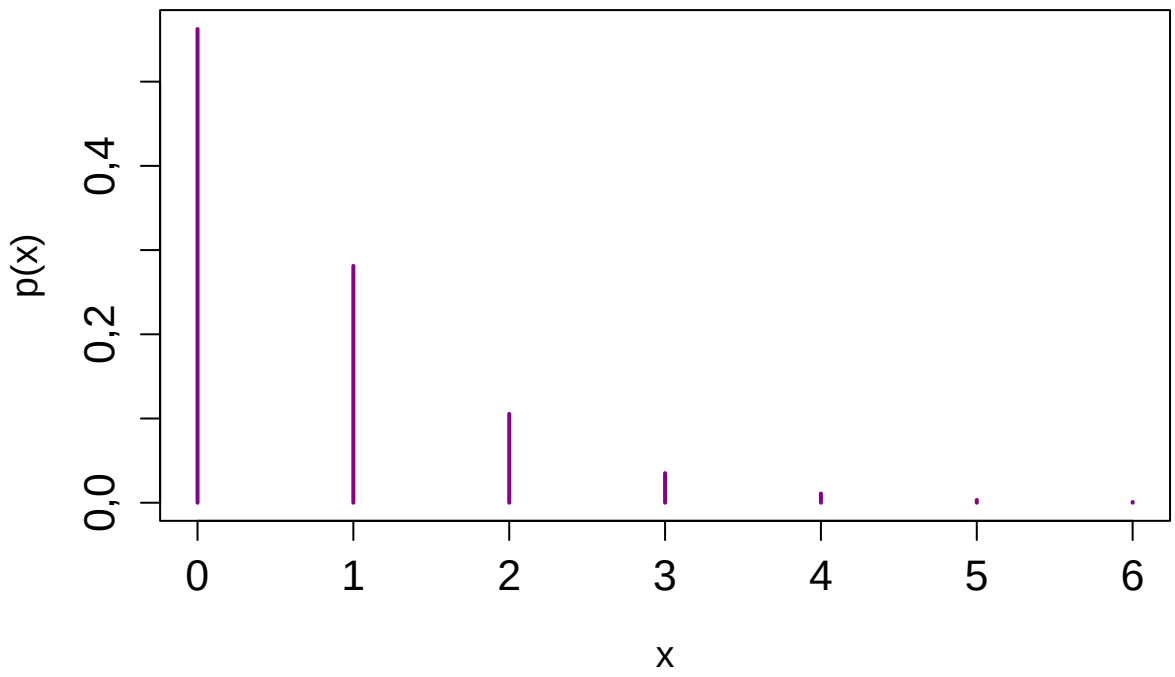


3. Binomial negativa ou Pascal

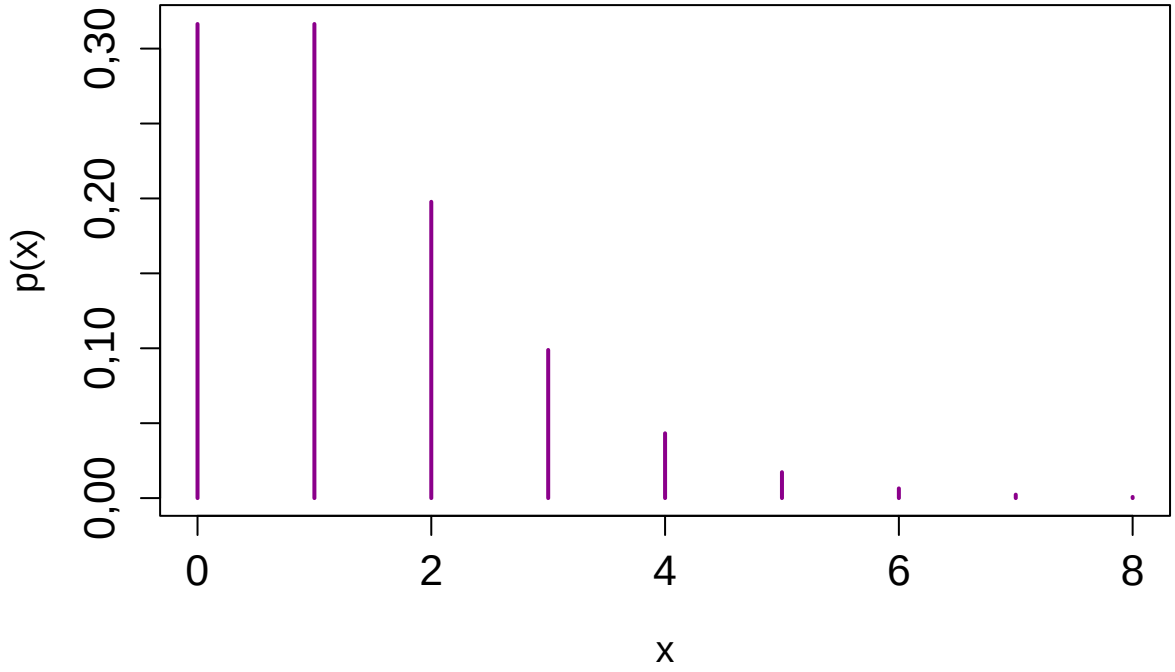
A variável aleatória X se refere ao número de falhas que antecedem os r primeiros sucessos, de modo que $x \in \{0, 1, 2, \dots\}$. Nas aulas utilizamos $X + r$, $r \geq 1$.

Sintaxe básica: `dnbinom(x, r, teta)`.

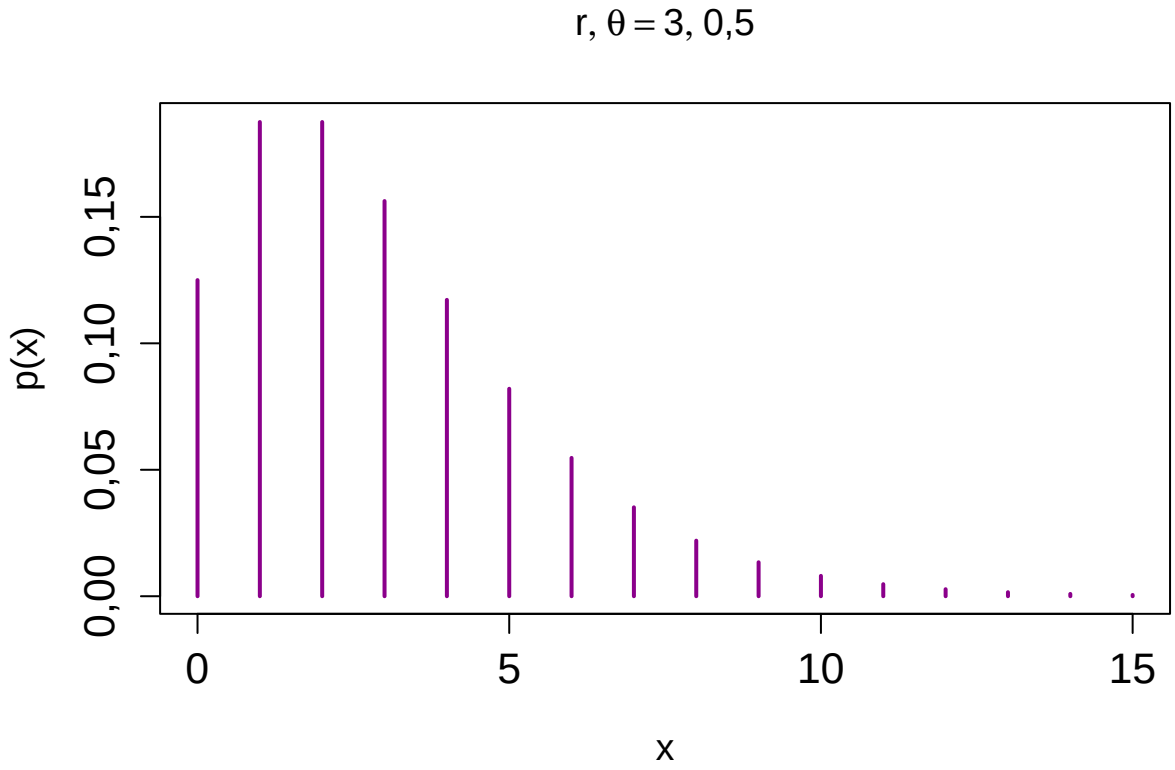
```
r <- 2L
teta <- 0.75
xmax <- qnbinom(0.999, r, teta)
curve(dnbinom(x, r, teta), from = 0, to = xmax, n = xmax + 1, type = "h",
      xlab = "x", ylab = "p(x)", col = "darkmagenta", lwd = 2,
      cex.main = 1.2, cex.axis = 1.3, cex.lab = 1.2,
      main = substitute(list(r, theta) == list(v1, v2),
             list(v1 = r, v2 = teta)))
```



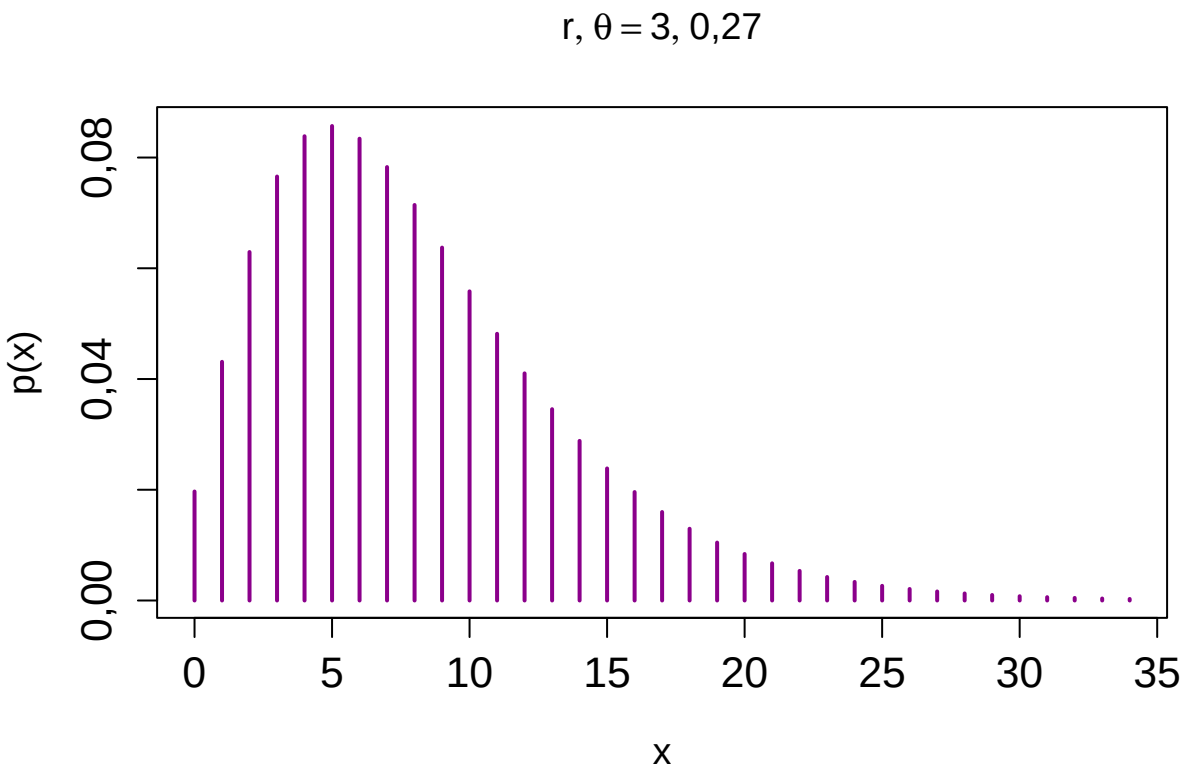

```
r <- 4L
teta <- 0.75
xmax <- qnbinom(0.999, r, teta)
curve(dnbinom(x, r, teta), from = 0, to = xmax, n = xmax + 1, type = "h",
      xlab = "x", ylab = "p(x)", col = "darkmagenta", lwd = 2,
      cex.main = 1.2, cex.axis = 1.3, cex.lab = 1.2,
      main = substitute(list(r, theta) == list(v1, v2),
             list(v1 = r, v2 = teta)))
```




```
r <- 3L
teta <- 0.5
xmax <- qnbinom(0.999, r, teta)
curve(dnbinom(x, r, teta), from = 0, to = xmax, n = xmax + 1, type = "h",
      xlab = "x", ylab = "p(x)", col = "darkmagenta", lwd = 2,
      cex.main = 1.2, cex.axis = 1.3, cex.lab = 1.2,
      main = substitute(list(r, theta) == list(v1, v2),
             list(v1 = r, v2 = teta)))
```



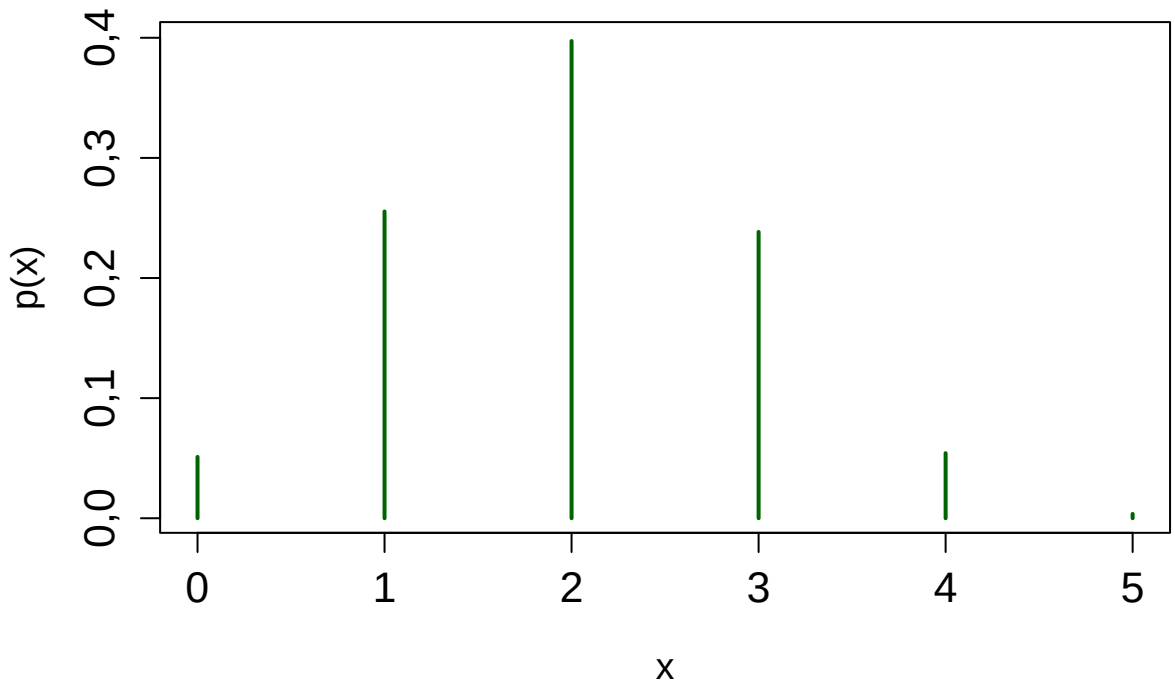

```
r <- 3L
teta <- 0.27
xmax <- qnbinom(0.999, r, teta)
curve(dnbinom(x, r, teta), from = 0, to = xmax, n = xmax + 1, type = "h",
      xlab = "x", ylab = "p(x)", col = "darkmagenta", lwd = 2,
      cex.main = 1.2, cex.axis = 1.3, cex.lab = 1.2,
      main = substitute(list(r, theta) == list(v1, v2),
             list(v1 = r, v2 = teta)))
```



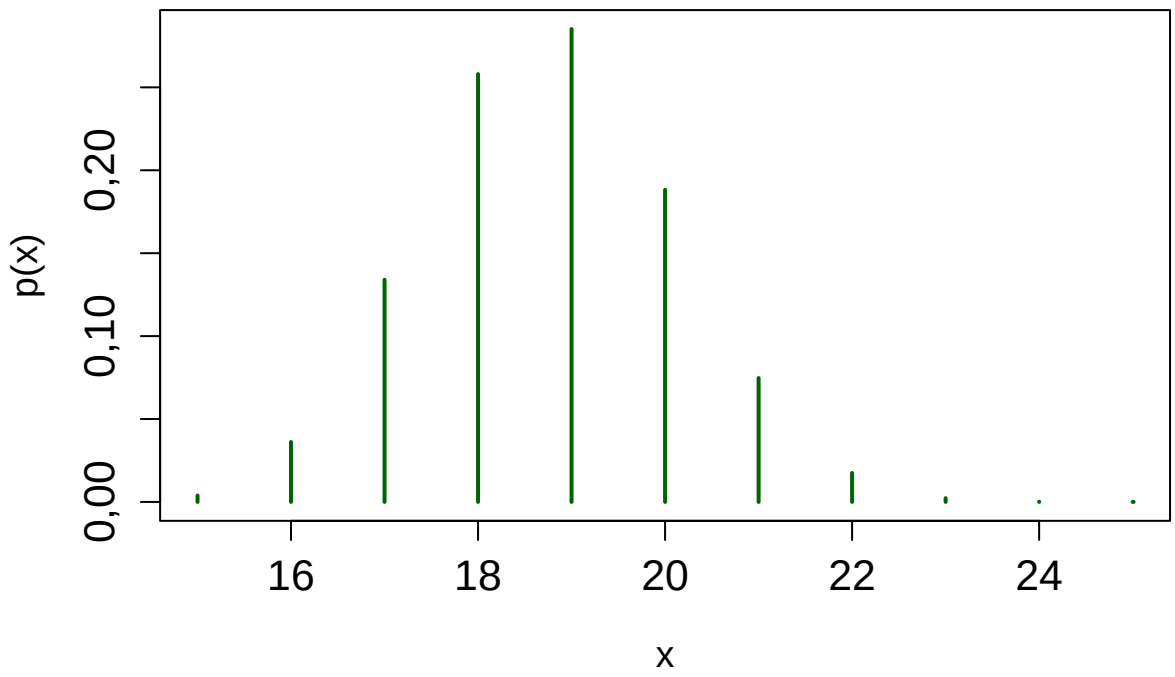
4. Hipergeométrica

Sintaxe básica: `dhyper(x, M, N - M, n)`.

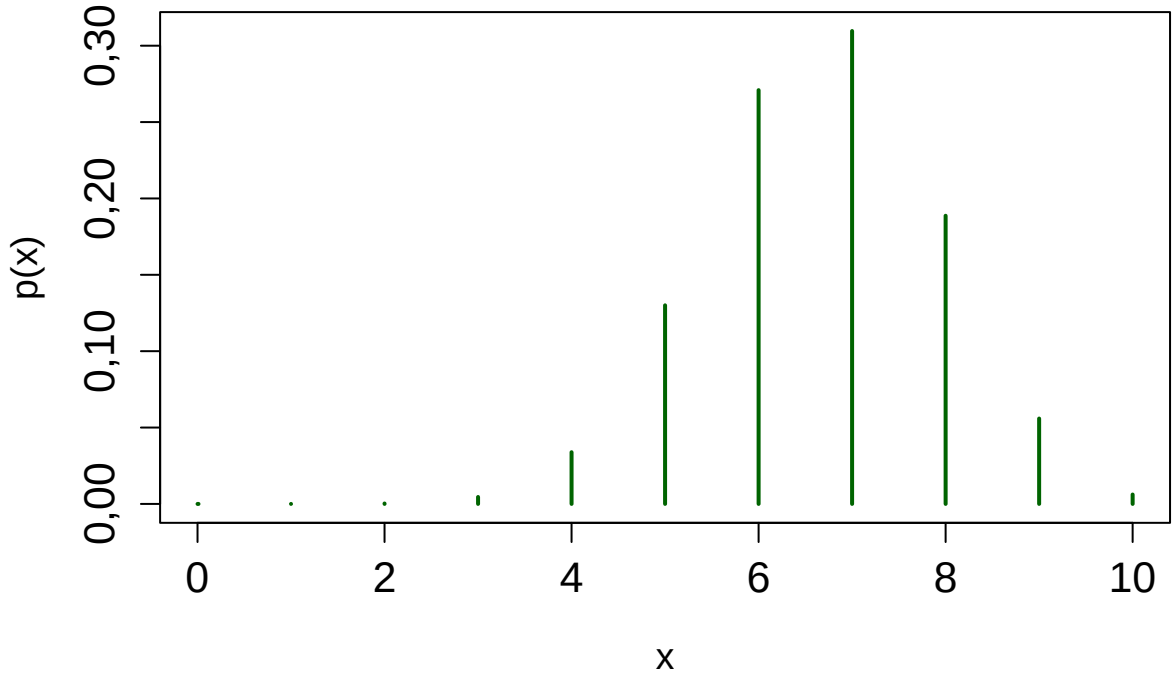
```
N <- 20
M <- 5
n <- 8
xmin <- max(0, n - N + M)
xmax <- min(n , M)
curve(dhyper(x, M, N - M, n), from = xmin, to = xmax,
     n = xmax - xmin + 1, lwd = 2, type = "h", xlab = "x",
     ylab = "p(x)", col = "darkgreen", cex.lab = 1.2,  cex.axis = 1.3,
     main = substitute(list(N, M, n) == list(v1, v2, v3),
     list(v1 = N, v2 = M, v3 = n)), cex.main = 1.2)
```



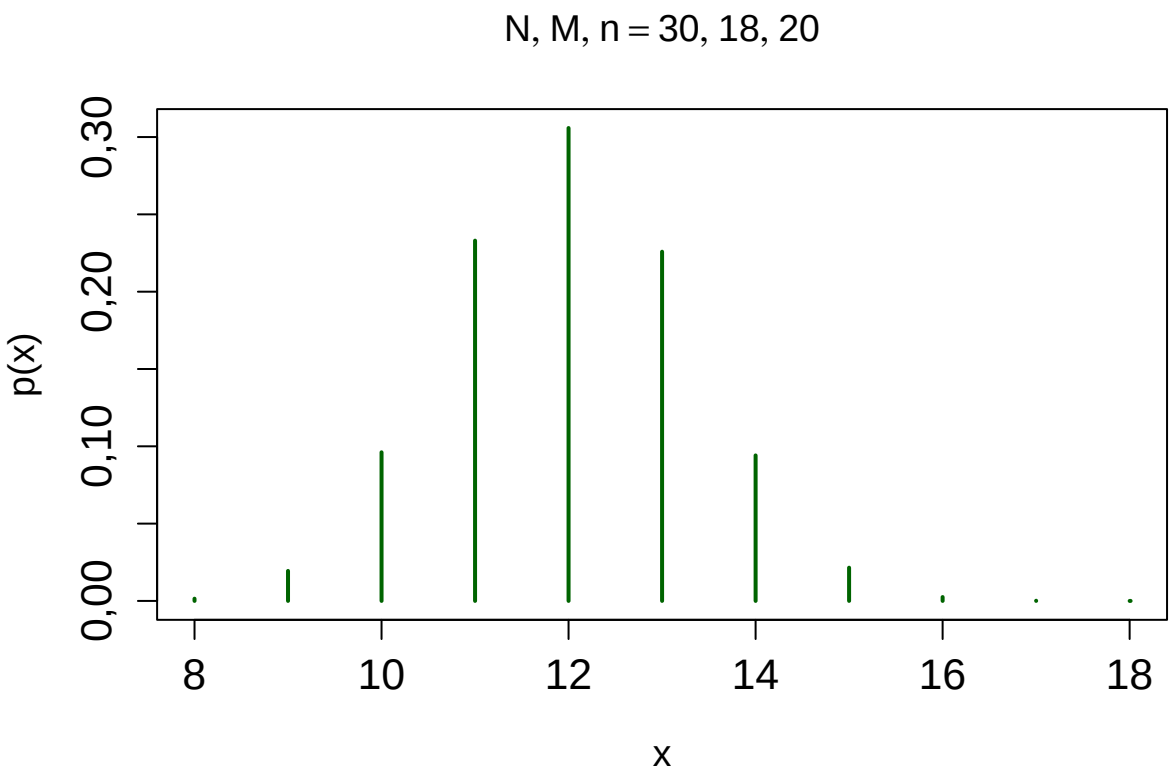

```r
N <- 40
M <- 25
n <- 30
xmin <- max(0, n - N + M)
xmax <- min(n , M)
curve(dhyper(x, M, N - M, n), from = xmin, to = xmax,
     n = xmax - xmin + 1, lwd = 2, type = "h", xlab = "x",
     ylab = "p(x)", col = "darkgreen", cex.lab = 1.2,  cex.axis = 1.3,
     main = substitute(list(N, M, n) == list(v1, v2, v3),
     list(v1 = N, v2 = M, v3 = n)), cex.main = 1.2)
```




```
N <- 30
M <- 20
n <- 10
xmin <- max(0, n - N + M)
xmax <- min(n , M)
curve(dhyper(x, M, N - M, n), from = xmin, to = xmax,
     n = xmax - xmin + 1, lwd = 2, type = "h", xlab = "x",
     ylab = "p(x)", col = "darkgreen", cex.lab = 1.2,  cex.axis = 1.3,
     main = substitute(list(N, M, n) == list(v1, v2, v3),
     list(v1 = N, v2 = M, v3 = n)), cex.main = 1.2)
```



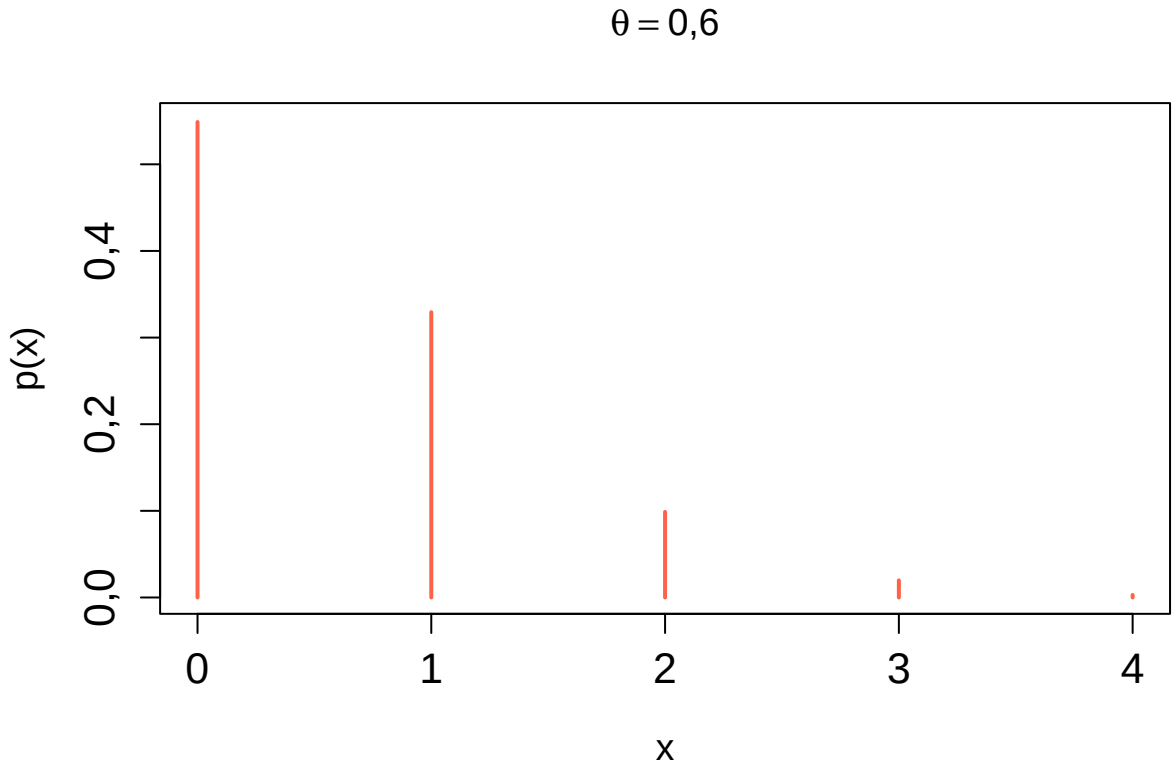

```
N <- 30
M <- 18
n <- 20
xmin <- max(0, n - N + M)
xmax <- min(n , M)
curve(dhyper(x, M, N - M, n), from = xmin, to = xmax,
     n = xmax - xmin + 1, lwd = 2, type = "h", xlab = "x",
     ylab = "p(x)", col = "darkgreen", cex.lab = 1.2,  cex.axis = 1.3,
     main = substitute(list(N, M, n) == list(v1, v2, v3),
     list(v1 = N, v2 = M, v3 = n)), cex.main = 1.2)
```



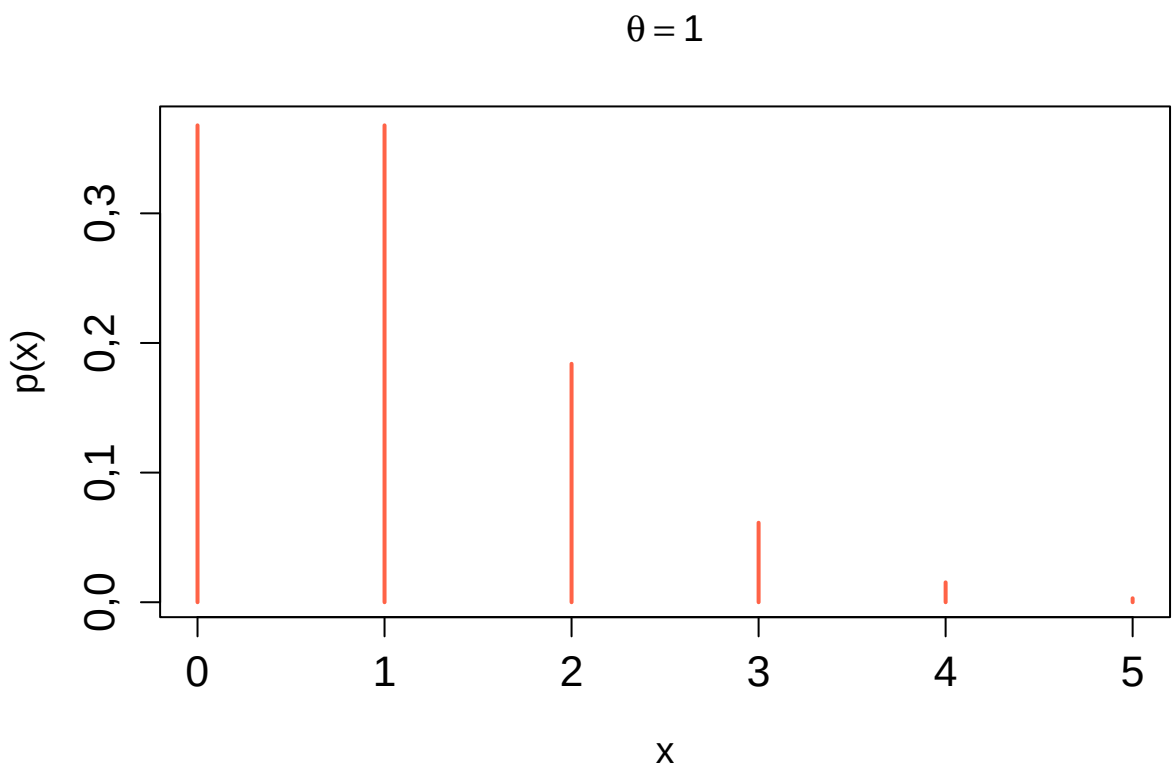
5. Poisson

Sintaxe básica: `dpois(x, teta)`.

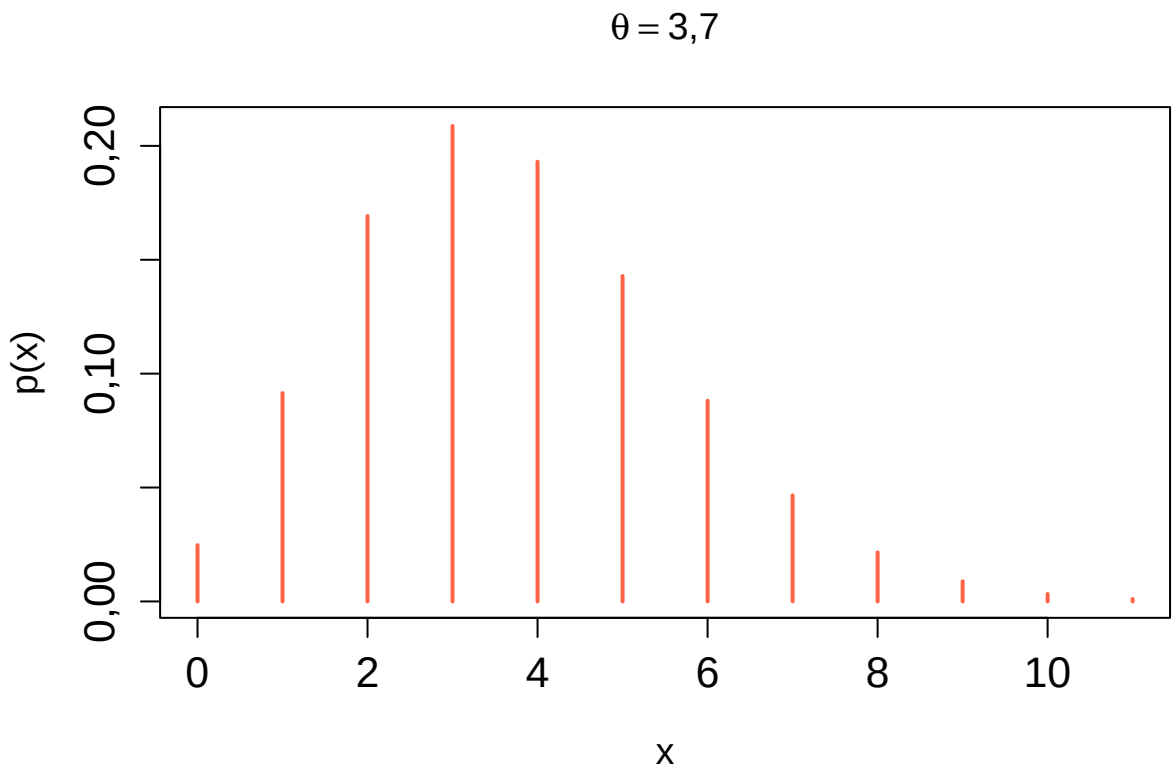
```
teta <- 0.6
xmax <- qpois(0.999, teta)
curve(dpois(x, teta), from = 0, to = xmax, n = xmax + 1, type = "h",
      lwd = 2, xlab = "x" , ylab = "p(x)", col = "tomato",
      main = substitute(list(theta) == list(v1), list(v1 = teta)),
      cex.main = 1.2,  cex.lab = 1.2, cex.axis = 1.3)
```



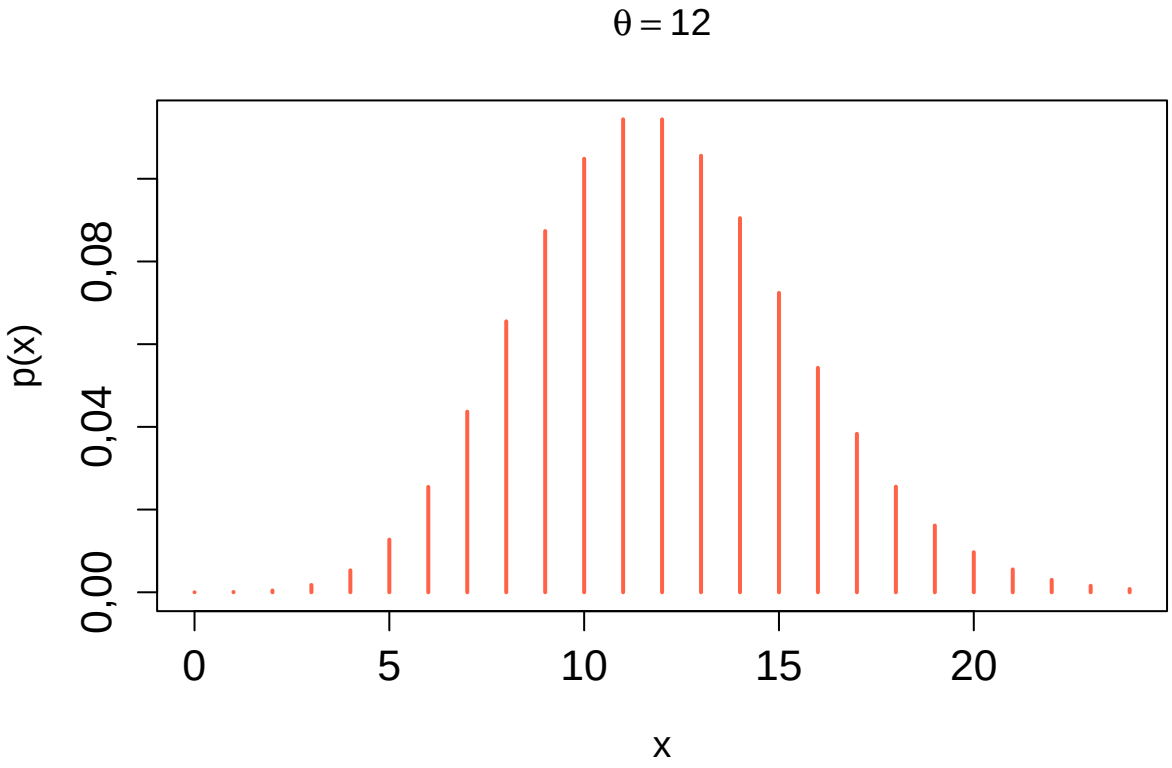

```
teta <- 1
xmax <- qpois(0.999, teta)
curve(dpois(x, teta), from = 0, to = xmax, n = xmax + 1, type = "h",
      lwd = 2, xlab = "x" , ylab = "p(x)", col = "tomato",
      main = substitute(list(theta) == list(v1), list(v1 = teta)),
      cex.main = 1.2,  cex.lab = 1.2, cex.axis = 1.3)
```




```
teta <- 3.7
xmax <- qpois(0.999, teta)
curve(dpois(x, teta), from = 0, to = xmax, n = xmax + 1, type = "h",
      lwd = 2, xlab = "x" , ylab = "p(x)", col = "tomato",
      main = substitute(list(theta) == list(v1), list(v1 = teta)),
      cex.main = 1.2,  cex.lab = 1.2, cex.axis = 1.3)
```




```r
teta <- 12
xmax <- qpois(0.999, teta)
curve(dpois(x, teta), from = 0, to = xmax, n = xmax + 1, type = "h",
      lwd = 2, xlab = "x" , ylab = "p(x)", col = "tomato",
      main = substitute(list(theta) == list(v1), list(v1 = teta)),
      cex.main = 1.2,  cex.lab = 1.2, cex.axis = 1.3)
```



6. Problema do colecionador de figurinhas

Figurinhas de um álbum são compradas uma a uma até que o álbum fique completo. O número de figurinhas é N e supomos que todas elas foram impressas em quantidades iguais. Considere a figurinha k, $k \in \{1, 2, \ldots, N\}$. A variável Y_k representa o número de figurinhas compradas após $k-1$ figurinhas já estarem no álbum. Sendo assim, $X = \sum_{k=1}^{N} Y_k$ representa o número de figurinhas compradas e Y_k tem distribuição geométrica com parâmetro

$$\theta_k = \frac{N-(k-1)}{N}.$$

Usando propriedades da distribuição geométrica, obtemos

$$\mu = E(X) = N\left(1 + \frac{1}{2} + \cdots + \frac{1}{N}\right) \quad \text{e} \quad \sigma^2 = \text{var}(X) = N^2\left(1 + \frac{1}{2^2} + \cdots + \frac{1}{N^2}\right).$$

Em um álbum com 300 figurinhas, média e desvio padrão são iguais a 1884,8 e 384,8, respectivamente.

Pode ser provado que

$$p(x) = \sum_{j=0}^{N-1} \binom{N-1}{j} \left(\frac{N-1-j}{N}\right)^{x-1}, \; x \in \{N, N+1, \ldots\}$$

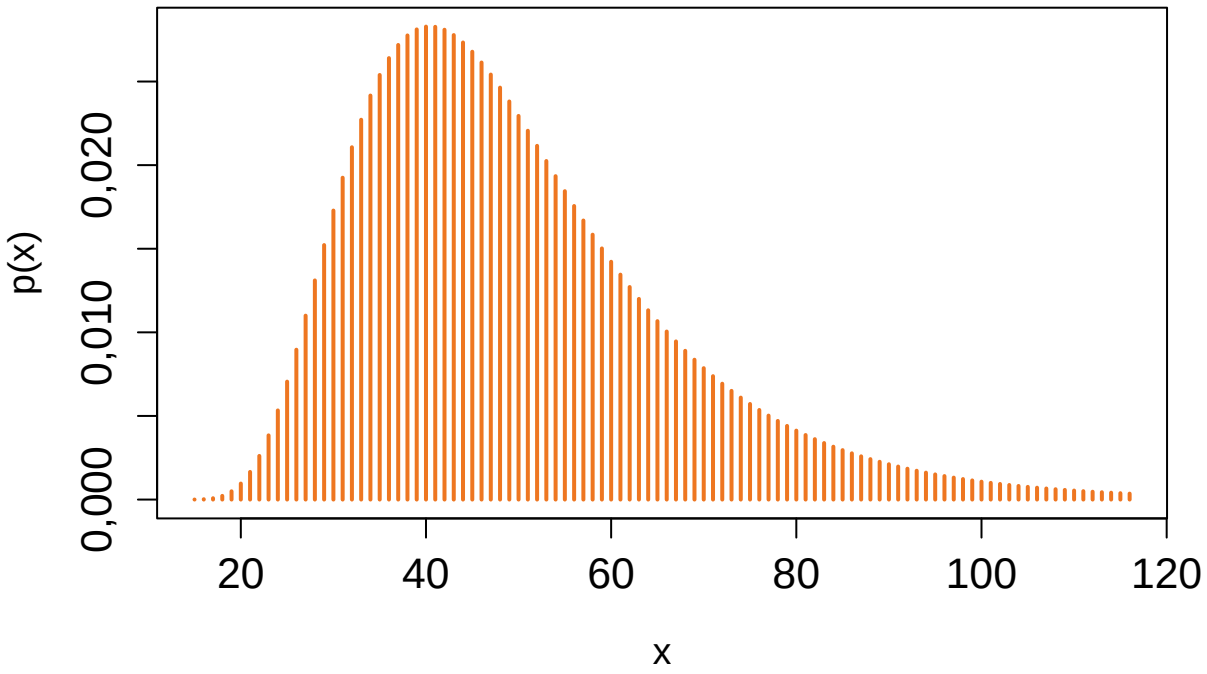
(vide https://stats.libretexts.org/Bookshelves/Probability_Theory/Probability_Mathematical_Statistics_and_Stochastic_Processes_(Siegrist)/12%3A_Finite_Sampling_Models/12.07%3A_The_Coupon_Collector_Problem).

```
# Se N é "grande", p(x) deve ser calculada de outra forma.
N <- 15L
```

```r
mu <- N * sum(1 / (1:N))
sig <- sqrt(N^2 * sum(1 / (1:N)^2))
cat("\n Média e desvio padrão:", c(mu, sig), "\n")
```

```
##
##  Média e desvio padrão: 49,77343 18,85733
```

```r
ns <- N:(round(mu + 3.5 * sig))
i <- 0
j <- 0:(N-1)
px <- c()
for (x in ns) {
  i <- i + 1
  px[i] <- sum((-1)^j * choose(N - 1, j) * ((N - 1 - j) / N)^(x - 1))
}
plot(ns, px, type = "h", lwd = 2, xlab = "x" , ylab = "p(x)",
     col = "chocolate2", cex.lab = 1.2, cex.axis = 1.3)
```

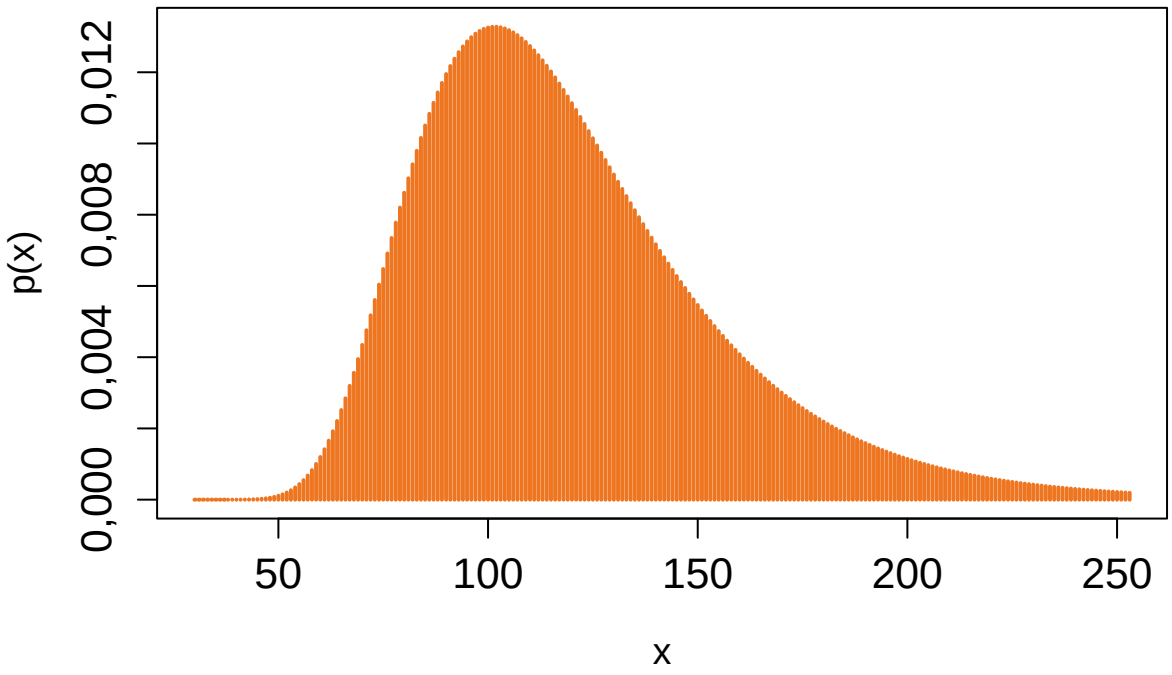



```r
# Se N é "grande", p(x) deve ser calculada de outra forma.
N <- 30L
mu <- N * sum(1 / (1:N))
sig <- sqrt(N^2 * sum(1 / (1:N)^2))
cat("\n Média e desvio padrão:", c(mu, sig), "\n")
```

```
##
##  Média e desvio padrão: 119,8496 38,09114
```

```r
ns <- N:(round(mu + 3.5 * sig))
i <- 0
j <- 0:(N-1)
px <- c()
for (nt in ns) {
  i <- i + 1
  px[i] <- sum((-1)^j * choose(N - 1, j) * ((N - 1 - j) / N)^(nt - 1))
```

```
}
plot(ns, px, type = "h", lwd = 2, xlab = "x" , ylab = "p(x)",
     col = "chocolate2", cex.lab = 1.2, cex.axis = 1.3)
```



7. Benford ou Newcomb-Benford

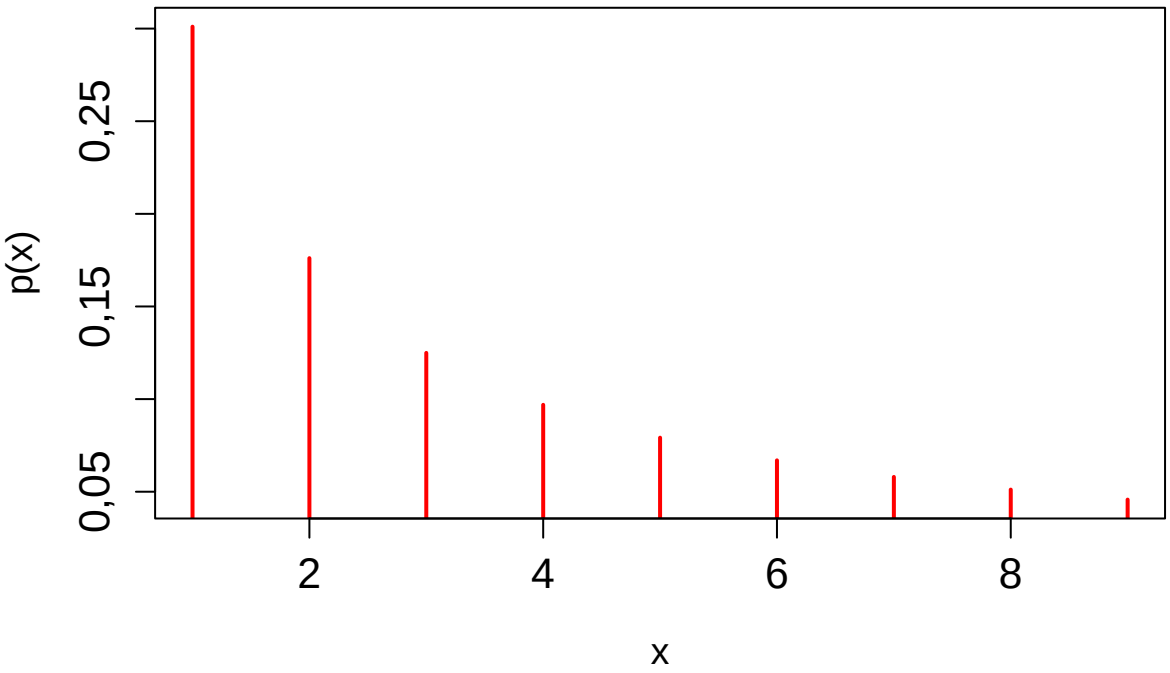
Foi estudada inicialmente por Newcomb (1881) e Benford (1938). Também chamada de lei do dígito significativo. Está implementada no pacote `VGAM`. Aplicações da distribuição são apresentadas no pacote `benford.analysis` (https://cran.r-project.org/web/packages/benford.analysis/benford.analysis.pdf). A função massa de probabilidade é

$$p(x) = \log_{10}\left(1 + \frac{1}{x}\right), \; x \in \{1, 2, \ldots, 9\},$$

notando que a distribuição não tem parâmetros.

Sintaxe básica: `dbenf(x)`.

```
library(VGAM)
curve(dbenf, from = 1, to = 9, n = 9, type = "h", lwd = 2,
      xlab = "x" , ylab = "p(x)", col = "red", cex.lab = 1.2,
      cex.axis = 1.3)
```



Nota 2. Refaça os exemplos em linguagem Python.

Nota 3. Diversas distribuições de probabilidade, tanto discretas quanto contínuas, estão disponíveis na linguagem R e estão descritas na página *CRAN Task View: Probability Distributions* (https://cran.r-project.org/web/views/Distributions.html).