Teste das sequências (ou corridas)

Linguagem R

2023

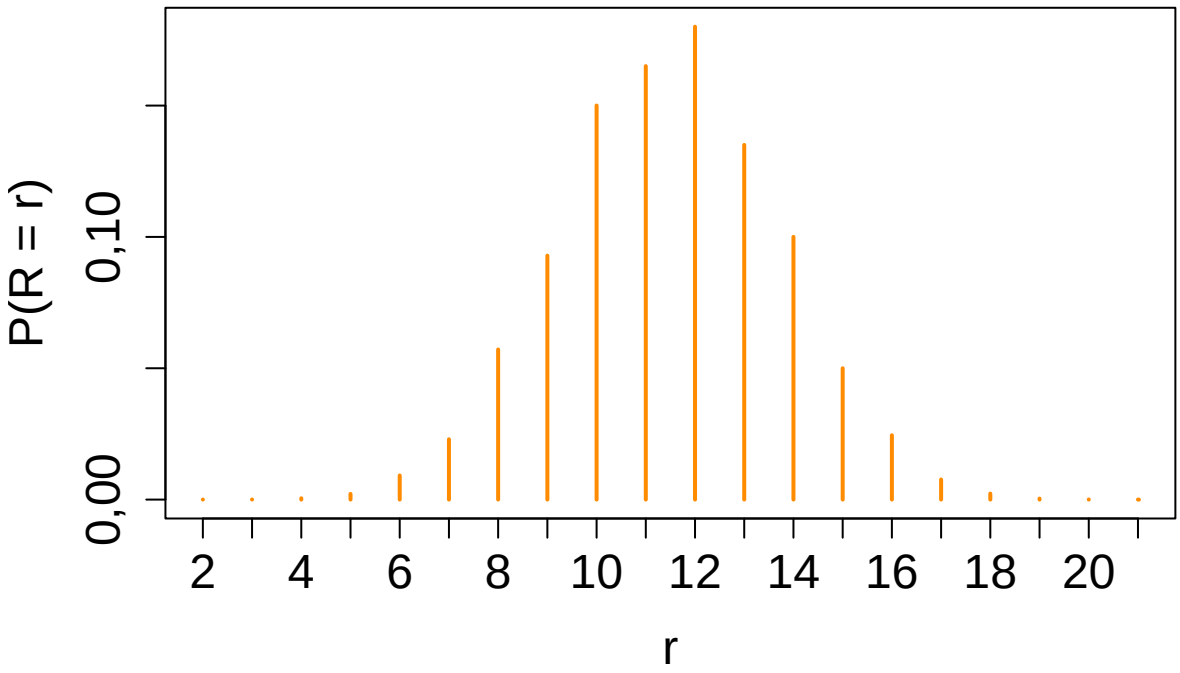
```
# Separador decimal: ","
options(OutDec = ",")
```

```
### Teste das sequências de Wald-Wolfowitz
library(randomizeBE)
```

Função massa de probabilidade

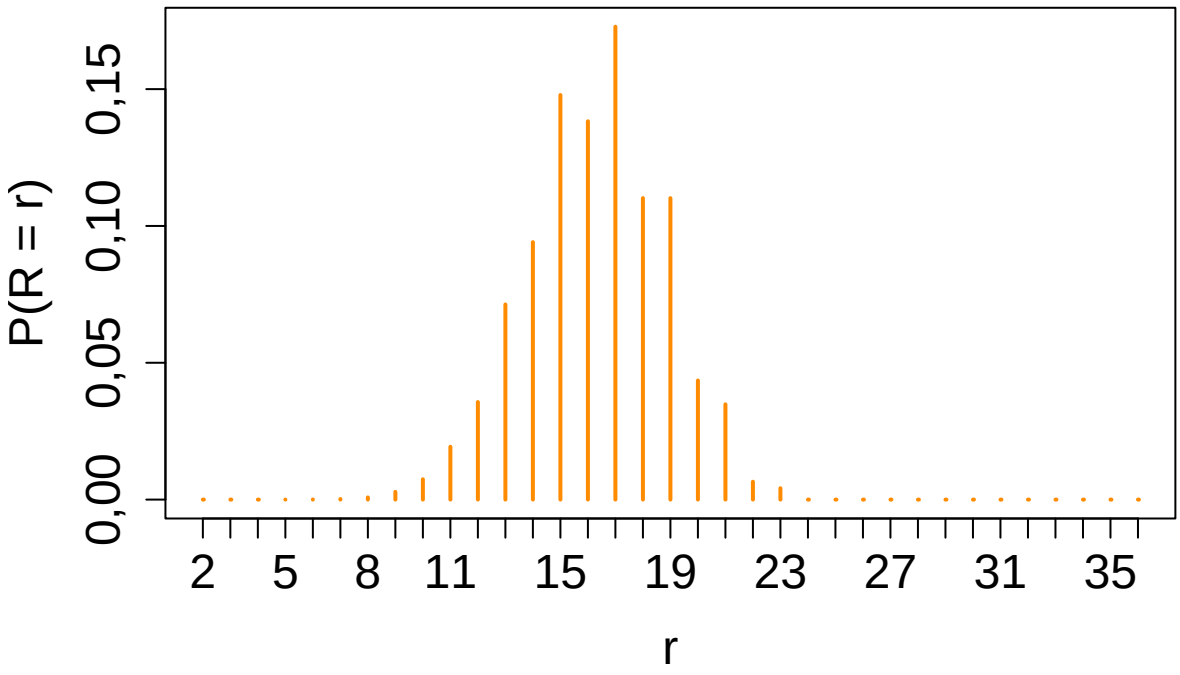
```
## Distribuição exata
n1 <- 11
n2 <- 10
n <- n1 + n2

# Função massa de probabilidade
# pruns.exact: função distribuição acumulada
fmp <- numeric(n - 2 + 1)
fda0 <- 0
for (r in 2:n) {
   fda <- pruns.exact(r, n1, n2, tail = "lower")
   fmp[r - 1] <- fda - fda0
   fda0 <- fda
}
```

```
plot(2:n, fmp, type = "h", xlab = "r", ylab = "P(R = r)", lwd = 2,
     axes = FALSE, cex.lab = 1.5, col = "darkorange")
axis(1, 2:n, cex.axis = 1.5)
axis(2, cex.axis = 1.5)
box()
```



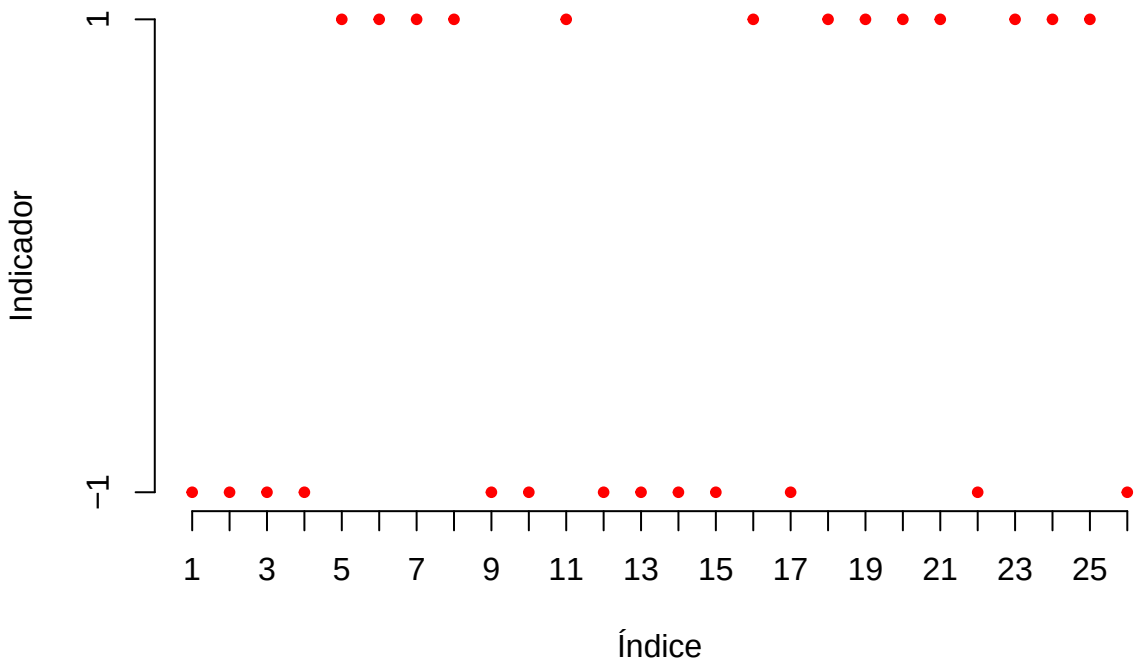

```r
## Distribuição exata
n1 <- 11
n2 <- 25
n <- n1 + n2

# Função massa de probabilidade
# pruns.exact: função distribuição acumulada
fmp <- numeric(n - 2 + 1)
fda0 <- 0
for (r in 2:n) {
   fda <- pruns.exact(r, n1, n2, tail = "lower")
   fmp[r - 1] <- fda - fda0
   fda0 <- fda
}
```

```r
plot(2:n, fmp, type = "h", xlab = "r", ylab = "P(R = r)", lwd = 2,
     axes = FALSE, cex.lab = 1.5, col = "darkorange")
axis(1, 2:n, cex.axis = 1.5)
axis(2, cex.axis = 1.5)
box()
```



Exemplo 1

```
# runs.pvalue: cálculo do valor-p para H1 bilateral
# Primeiro argumento deve ser um vetor numérico
# Dicotomização com a mediana se existirem mais de dois valores diferentes
# -1: menor do que a mediana e +1: maior do que ou igual à mediana

## x ~ N(0, 1)
set.seed(56)
x <- rnorm(27)

md <- median(x)
y <- ifelse(x >= md, 1, -1)
ind <- which(x == md)
if (length(ind) > 0) y <- y[-ind]

plot(y, pch = 20, axes = FALSE, xlab = "Índice", ylab = "Indicador",
     col = "red")
axis(1, 1:length(y))
axis(2, c(-1, 1))
```



Ressalte-se que "if `pmethod="exact"` is chosen and $n > 30$ and $n_1 > 12$ and $n_2 > 12$, the continuity corrected version of the normal approximation is used to save time and memory.".

```
runs.pvalue(x, pmethod = "exact")
```

```
## [1] 0,237744
```

```
# Aproximação sem correção de continuidade
runs.pvalue(x, pmethod = "normal")
```

```
## [1] 0,1711516
```

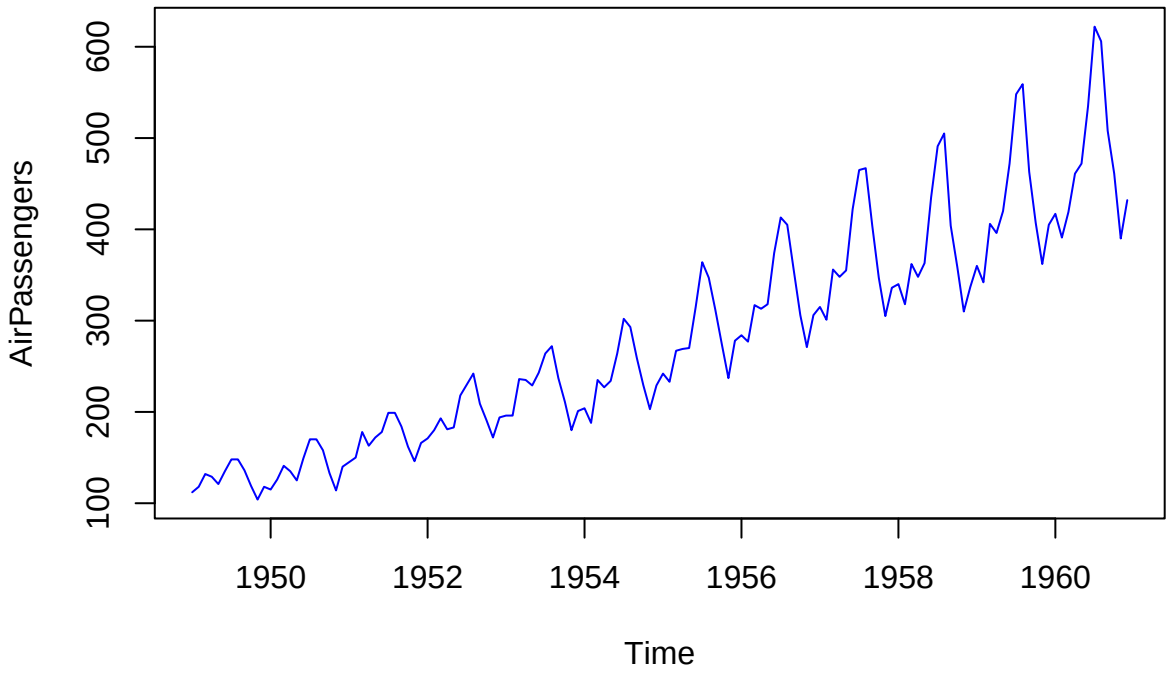
```
# Aproximação com correção de continuidade
runs.pvalue(x, pmethod = "cc")
```

```
## [1] 0,2412086
```

Nota 1. Qual o número de corridas neste exemplo?

Exemplo 2

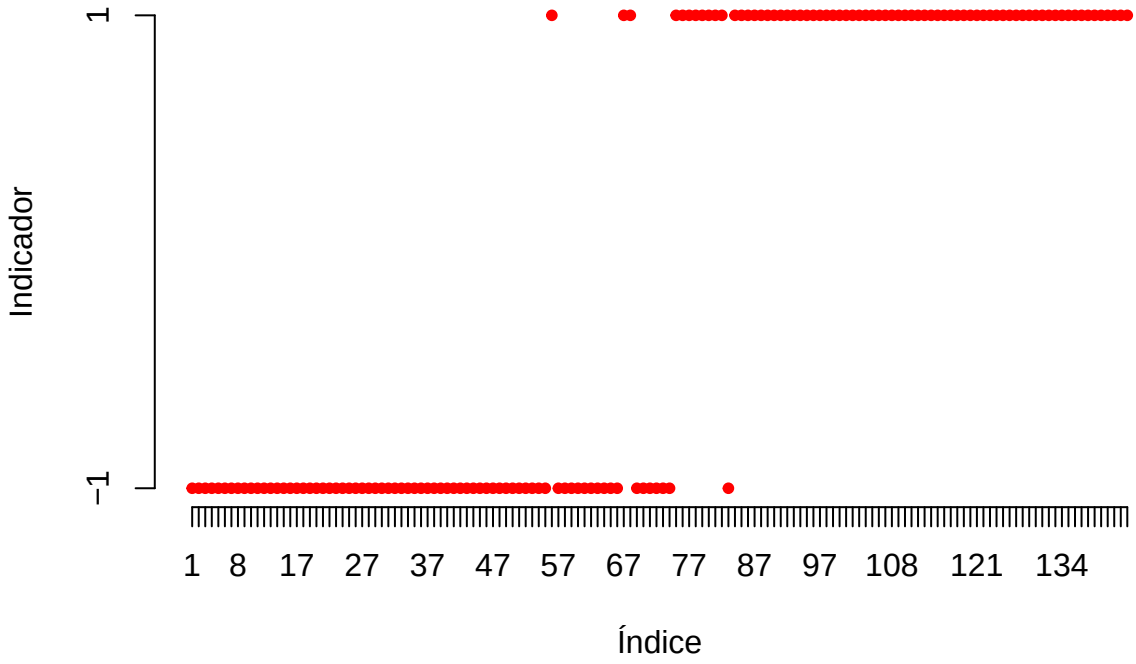
```
## Monthly airline passenger numbers 1949-1960 (n = 144)
# Conjunto de dados AirPassengers do pacote datasets

plot(AirPassengers, col = "blue")
```




```
x <- as.numeric(AirPassengers)
cat("\n n = ", length(x))
```

```
##
##  n =  144
```

```
md <- median(x)
y <- ifelse(x >= md, 1, -1)
ind <- which(x == md)
if (length(ind) > 0) y <- y[-ind]
plot(y, pch = 20, axes = FALSE, xlab = "Índice", ylab = "Indicador",
     col = "red")
axis(1, 1:length(y))
axis(2, c(-1, 1))
```




```
runs.pvalue(AirPassengers, pmethod = "exact")
```

```
## [1] 3,931479e-27
```

```
runs.pvalue(AirPassengers, pmethod = "normal")
```

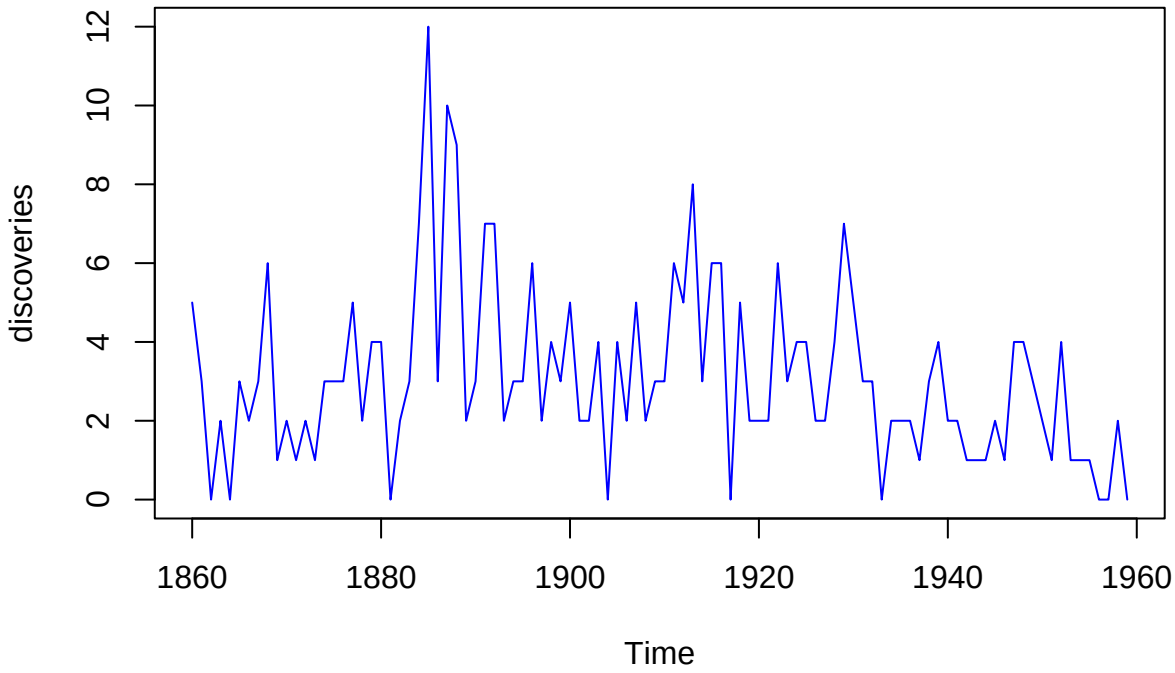
```
## [1] 1,577406e-27
```

```
runs.pvalue(AirPassengers, pmethod = "cc")
```

```
## [1] 3,931479e-27
```

Exemplo 3

```
# The numbers of "great" inventions and scientific discoveries
#         in each year from 1860 to 1959 (n = 100)
# Conjunto de dados discoveries do pacote datasets

plot(discoveries, col = "blue")
```



Existem observações com valor igual à mediana amostral.

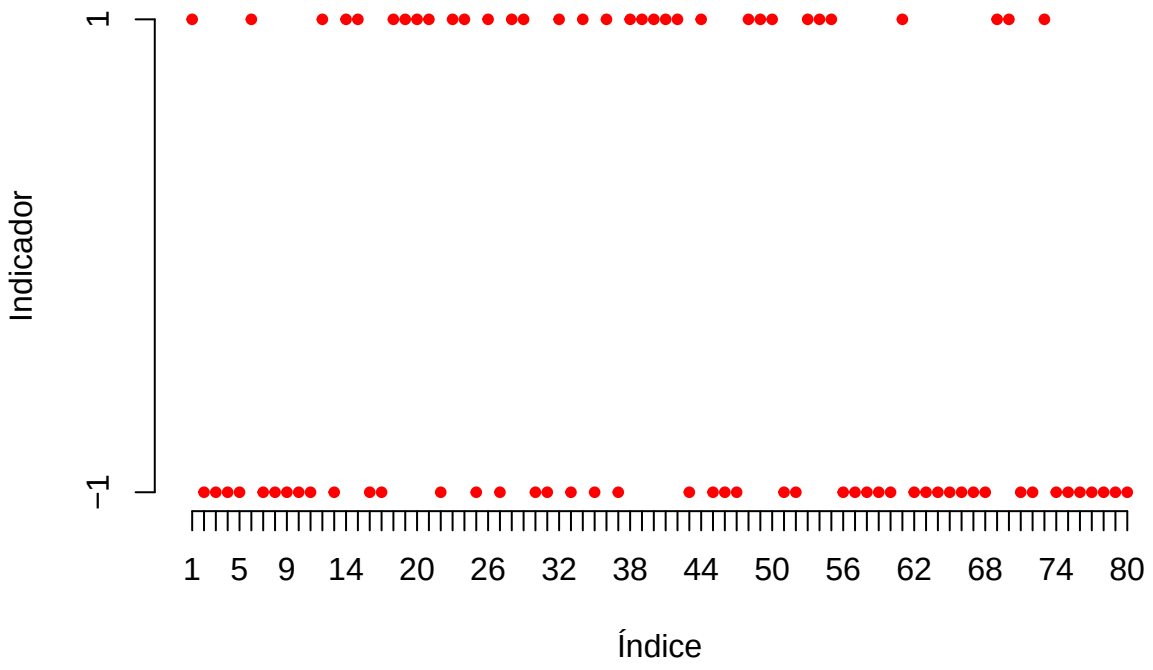
```
x <- as.numeric(discoveries)
md <- median(x)
cat("\n Mediana =", md)
```

```
##
##  Mediana = 3
```

```
(ind <- which(x == md))
```

```
##  [1]  2  6  8 15 16 17 24 27 31 35 36 40 50 51 55 64 72 73 79 90
```

```
y <- ifelse(x >= md, 1, -1)
if (length(ind) > 0) y <- y[-ind]
plot(y, pch = 20, axes = FALSE, xlab = "Índice", ylab = "Indicador",
     col = "red")
axis(1, 1:length(y))
axis(2, c(-1, 1))
```



A função `runs.pvalue` não elimina as observações com valor igual à mediana amostral.

```
runs.pvalue(discoveries, pmethod = "exact")
```

```
## [1] 0,01293409
```

```
runs.pvalue(discoveries, pmethod = "normal")
```

```
## [1] 0,009696892
```

```
runs.pvalue(discoveries, pmethod = "cc")
```

```
## [1] 0,01293409
```

Aplicando o teste à variável dicotomizada (`y`) obtemos

```
runs.pvalue(y, pmethod = "exact")
```

```
## [1] 0,4469061
```

```
runs.pvalue(y, pmethod = "normal")
```

```
## [1] 0,3806481
```

```
runs.pvalue(y, pmethod = "cc")
```

```
## [1] 0,4469061
```