Processos Estocásticos (Online)

Lucas Bicalho

03/09/2020

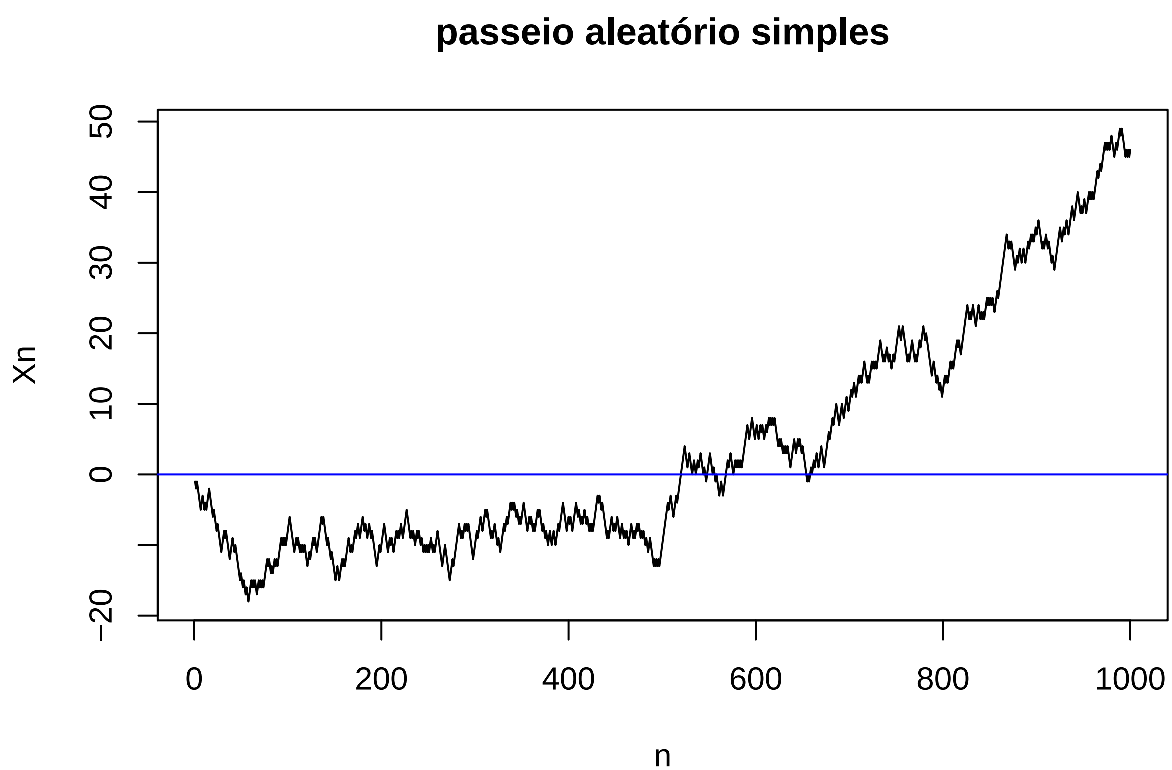
Passeio Aleatório Simples

Espaço de estados discreto a tempo discreto

```
#p(x=1)=p e p(x=-1)=1-p
p <- 0.5
q <- 1-p
n <- 1000
x <- ifelse(purrr::rbernoulli(n = n, p = p) == TRUE, 1, -1)
head(x, 10)
```

```
##  [1] -1 -1  1 -1 -1 -1 -1  1  1 -1
```

```
#{Sn, n>=0} = X1+ X2 + ... + Xn
# E = inteiros
# T = {0, 1, 2, ...}
sn <- cumsum(x)
plot(sn,
     type = "l",
     main = "passeio aleatório simples",
     ylab = "Xn",
     xlab = "n")
abline(h=0, col = 4)
```



Processo de Contagem

Espaço de estado discreto a tempo discreto ou tempo contínuo

```
#{Xt, t>=0}
# X0 = 0
# Xt >= 0
# Xt é valor inteiro
# Xs <= Xt se s < t
# Xt - Xs = nº de ocorrências em (s,t]
```

Processo de Poisson

Processo de Bernoulli

$\{X_n\}_{n\geq 1}$ processo a tempo discreto tal que as variáveis aleatórias X_i são i.i.d. com $P(X_n = 1) = p$ e $P(X_n = 0) = 1 - p$ é chamado processo de Bernoulli com parâmetro p.

Para cada realização do processo o valor é TRUE ou FALSE.

```
p <- 0.5
n <- 1000
```

```r
x <- purrr::rbernoulli(n = n, p = p)
head(x, 10)
```

```
##  [1] FALSE  TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE FALSE FALSE
```

```r
mean(x)
```

```
## [1] 0.481
```

```r
var(x)
```

```
## [1] 0.2498889
```

```r
p*(1-p)
```

```
## [1] 0.25
```

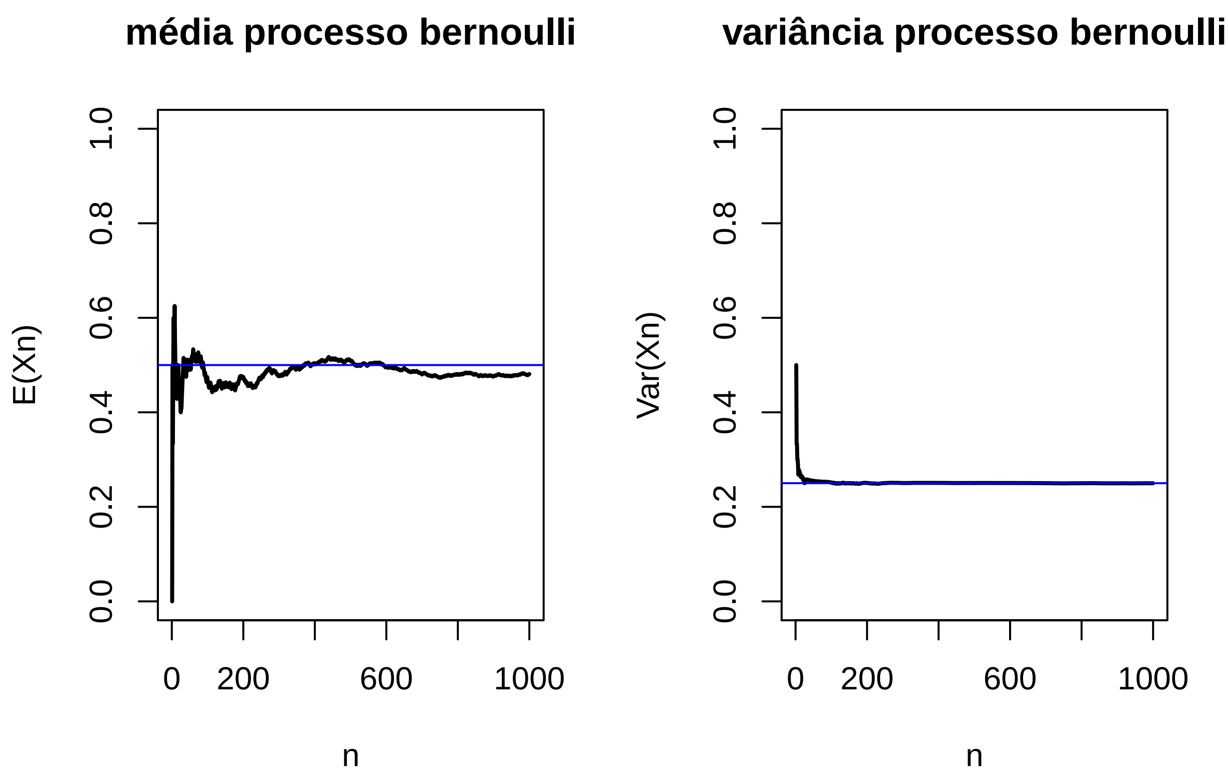
Vamos observar como se comportam a média e variância no decorrer dos ensaios

```r
mean_x <- c()
var_x <- c()

for(i in 1:n){
  mean_x[i] <- mean(x[1:i])
  var_x[i] <- var(x[1:i])
}

par(mfrow = c(1,2))
plot(mean_x,
     type = "l",
     main = "média processo bernoulli",
     xlab = "n",
     ylab = "E(Xn)",
     ylim = c(0,1),
     lwd = 2)
abline(h=p, col = 4)

plot(var_x,
     type = "l",
     main = "variância processo bernoulli",
     xlab = "n",
     ylab = "Var(Xn)",
     ylim = c(0,1),
     lwd = 2)
abline(h=p*(1-p), col = 4)
```




```
par(mfrow = c(1,1))
```

Processo Binomial

Se definirmos $\{S_n\}_{n\geq 1}$ como $S_n = \sum_{i=1}^{n} X_i$, $i = 1, \ldots, n$ com $n \geq 1$, temos que

-$S_n \sim Bin(n, p)$

-$E(S_n) = np$

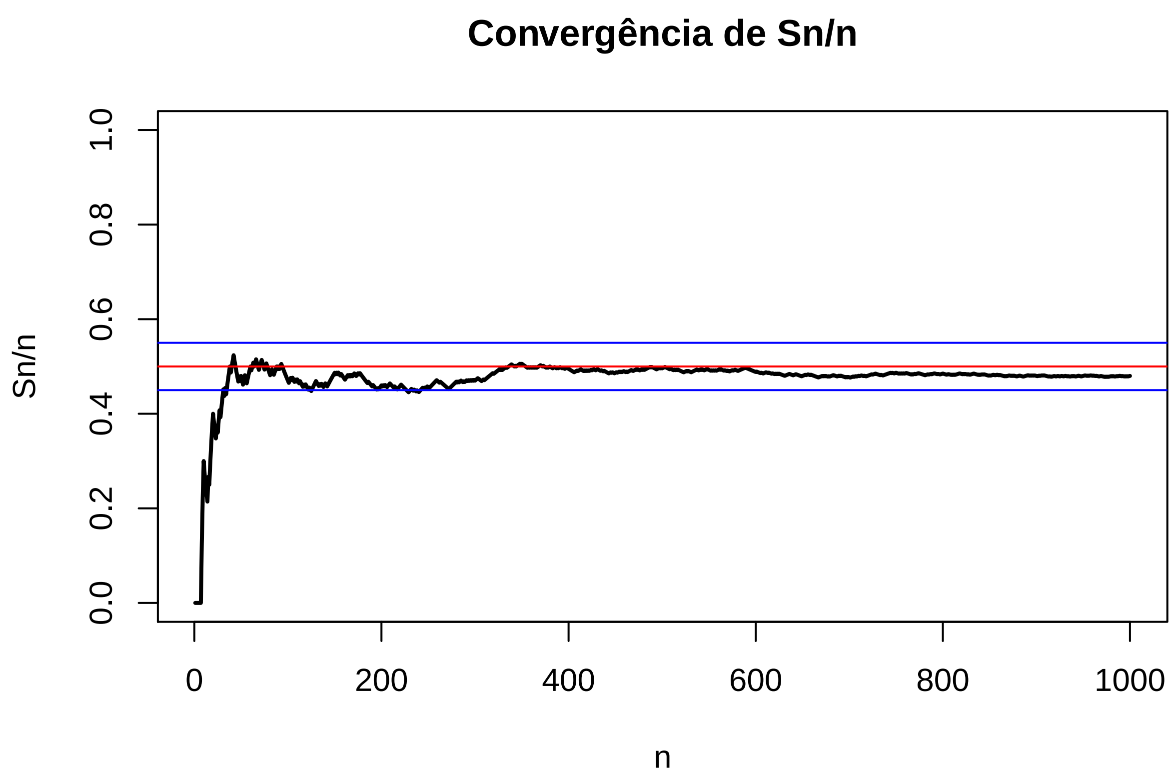
-$Var(S_n) = np(1-p)$

A Lei Forte dos Grandes Números diz que S_n/n converge quase certamente para p. Onde $S_n/n \in [p-\epsilon; p+\epsilon]$

```
p <- 0.5
n <- 1000
e <- 0.05
x <- purrr::rbernoulli(n = n, p = p)
x <- cumsum(x)
s <- 1:n
s <- x/s

plot(s,
     type = "l",
     main = "Convergência de Sn/n",
     xlab = "n",
```

```
      ylab = "Sn/n",
      ylim = c(0,1),
      lwd = 2)
abline(h=p-e, col = 4)
abline(h=p+e, col = 4)
abline(h=p, col = 2)
```



Processo de Chegadas

Seja $\{X_n\}_{n\geq 1}$ o processo de Bernoulli com parâmetro p. Para $n \geq 1$ defina T_n como o instante do n-ésimo sucesso.

$\{T_n\}_{n\geq 1}$ é chamado processo de chegadas.

Processo entre chegadas

Seja $W_n = T_n - T_{n-1}$. W_n é o tempo entre o (n-1)-ésimo e o n-ésimo sucesso.

O processo $\{W_n\}_{n\geq 1}$ é chamado processo entre chegadas.

Teorema

- $T_n \sim Pascal(n, p)$

- $E(T_n) = \frac{n}{p}$
- $Var(T_n) = \frac{n(1-p)}{p^2}$

* $W_n \sim Geométrica(p)$
  - $E(W_n) = \frac{1}{p}$
  - $Var(W_n) = \frac{1-p}{p^2}$

Distribuição Pascal (Binomial negativa): número de tentativas necessárias para obter k sucessos ao fim de n ensaios de $Bernoulli(p)$.

Distribuição Geométrica: número de tentativas necessárias para atingir o primeiro sucesso.

Lei Forte dos Grandes Números

$\frac{T_n}{n}$ converge quase que certamente para $\frac{1}{p}$ Note que $\frac{T_n}{n} = \sum_{i=1}^{n} \frac{w_i}{n}$

```r
p <- 0.5
n <- 10000
e <- 0.5
x <- purrr::rbernoulli(n = n, p = p)
t <- c()
k <- 1
for(i in 1:n){
  # t é o instante do i-ésimo sucesso
  if(x[i]==TRUE){
    t[k] <- i
    k <- k + 1
  }
}

head(t,10)
```

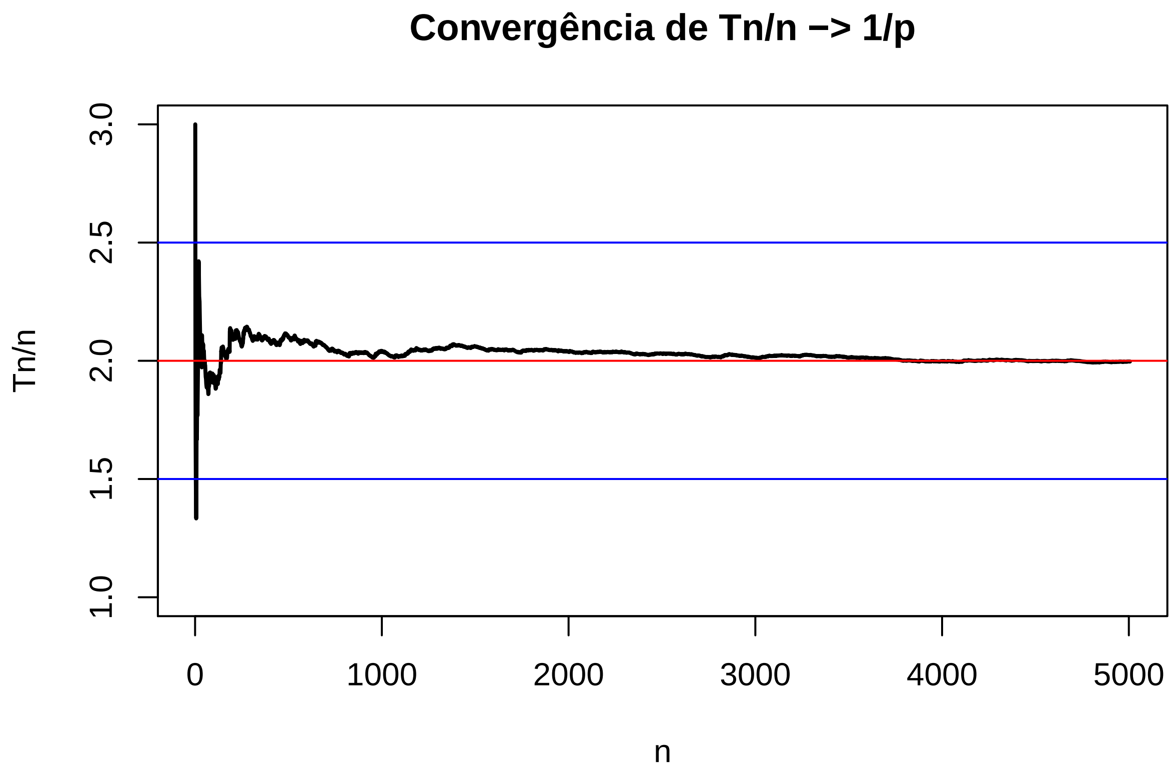
```
##  [1]  3  4  5  6  7  8 10 14 15 17
```

Ou seja, precisou-se de 17 ensaios de Bernoulli para obter-se o 10º sucesso.

Para $p = 0.5$ temos $\frac{1}{p} = \frac{1}{0.5} = 2$.

```r
tt <- 1:length(t)
tt <- t/tt

plot(tt,
     type = "l",
     main = "Convergência de Tn/n -> 1/p",
     xlab = "n",
     ylab = "Tn/n",
     lwd = 2,
     ylim = c((1/p)-1, (1/p)+1))
abline(h=(1/p)-e, col = 4)
abline(h=(1/p)+e, col = 4)
abline(h=1/p, col = 2)
```


Convergência de Tn/n −> 1/p
Tn/n
3.0
2.5
2.0
1.5
1.0
0
1000
2000
3000
4000
5000
n