Distribucion geometrica

Victor Lopez

2023-02-21

Si X es variable aleatoria que mide el "número de fracasos del experimento hasta antes de haber conseguido éxito", diremos que X se distribuye como una Geométrica con parámetro p

$$X \sim \text{Ge}(p)$$

Si X es variable aleatoria que mide el "número de repeticiones independientes del experimento hasta haber conseguido éxito", diremos que X se distribuye como una Geométrica con parámetro p

$$X \sim \text{Ge}(p)$$

- La **función de distribución** vendrá dada por

$$F(x) = \begin{cases} 0 & \text{si } x < 0 \\ 1-(1-p)^{k+1} & \text{si } k \leq x < k+1,\ k \in \mathbb{N} \end{cases}$$

- **Esperanza** $E(X) = \frac{1-p}{p}$ si empieza en 0 y $\text{E}(X) = \frac{1}{p}$ si empieza en 1
- **Varianza** $Var(X) = \frac{1-p}{p^2}$

El código de la distribución Geométrica:

- En `R` tenemos las funciones del paquete `Rlab`: `dgeom(x, prob)`, `pgeom(q, prob)`, `qgeom(p, prob)`, `rgeom(n, prob)` donde `prob` es la probabilidad de éxito del experimento.
- En `Python` tenemos las funciones del paquete `scipy.stats.geom`: `pmf(k,p)`, `cdf(k,p)`, `ppf(q,p)`, `rvs(p, size)` donde `p` es la probabilidad de éxito del experimento.

En R

```
library(Rlab)
```

```
## Rlab 4.0 attached.
```

```
##
## Attaching package: 'Rlab'
```

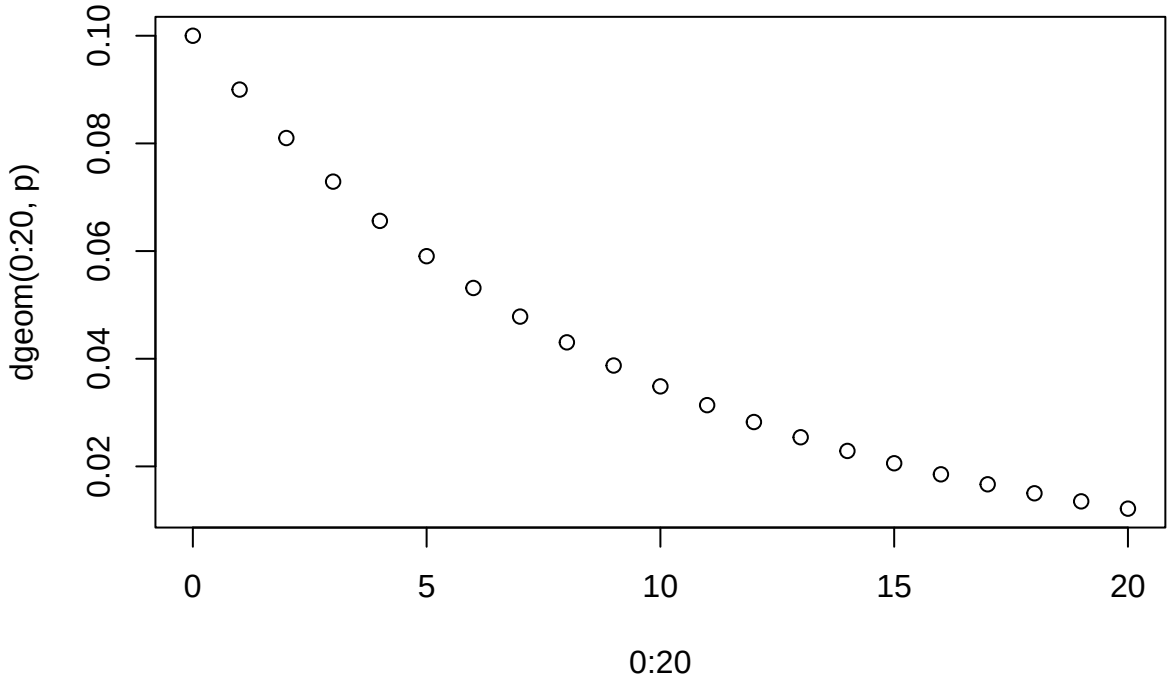
```
## The following objects are masked from 'package:stats':
##
##     dexp, dgamma, dweibull, pexp, pgamma, pweibull, qexp, qgamma,
##     qweibull, rexp, rgamma, rweibull
```

```
## The following object is masked from 'package:datasets':
##
##     precip
```

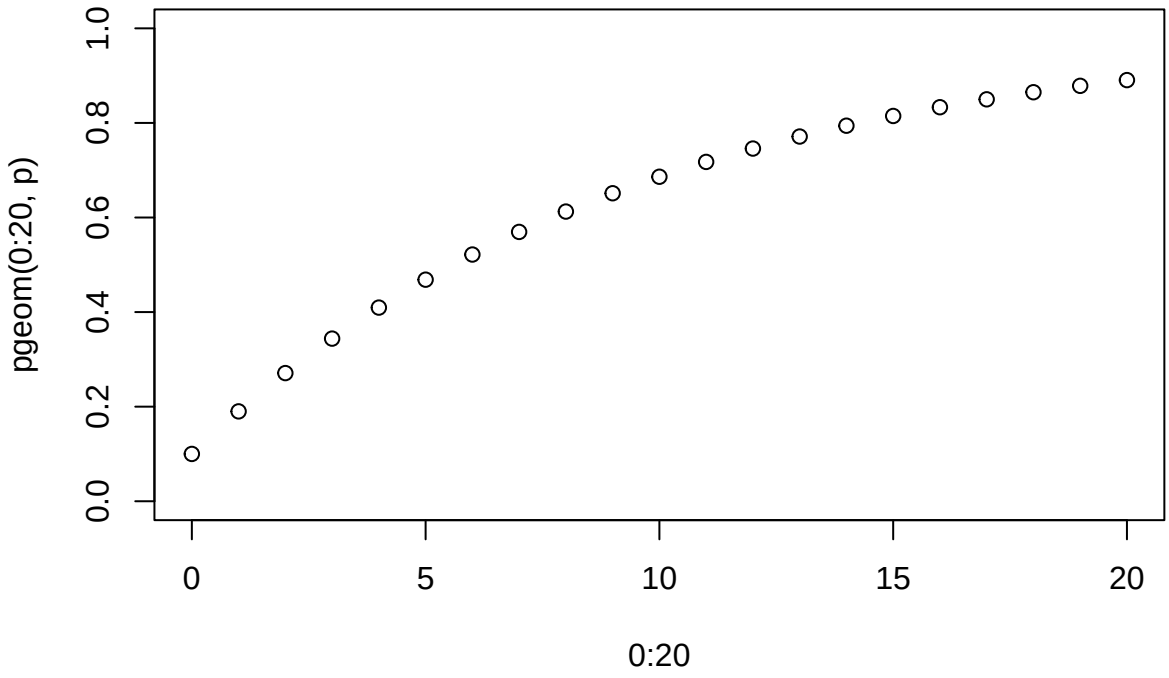
```
p = 0.1
dgeom(0:10, p) # Devuelve las probabilidades de 0 a 10 fracasos antes del exito
```

```
##  [1] 0.10000000 0.09000000 0.08100000 0.07290000 0.06561000 0.05904900
##  [7] 0.05314410 0.04782969 0.04304672 0.03874205 0.03486784
```

```
plot(0:20, dgeom(0:20, p))
```




```
plot(0:20, pgeom(0:20, p), ylim = c(0,1))
```



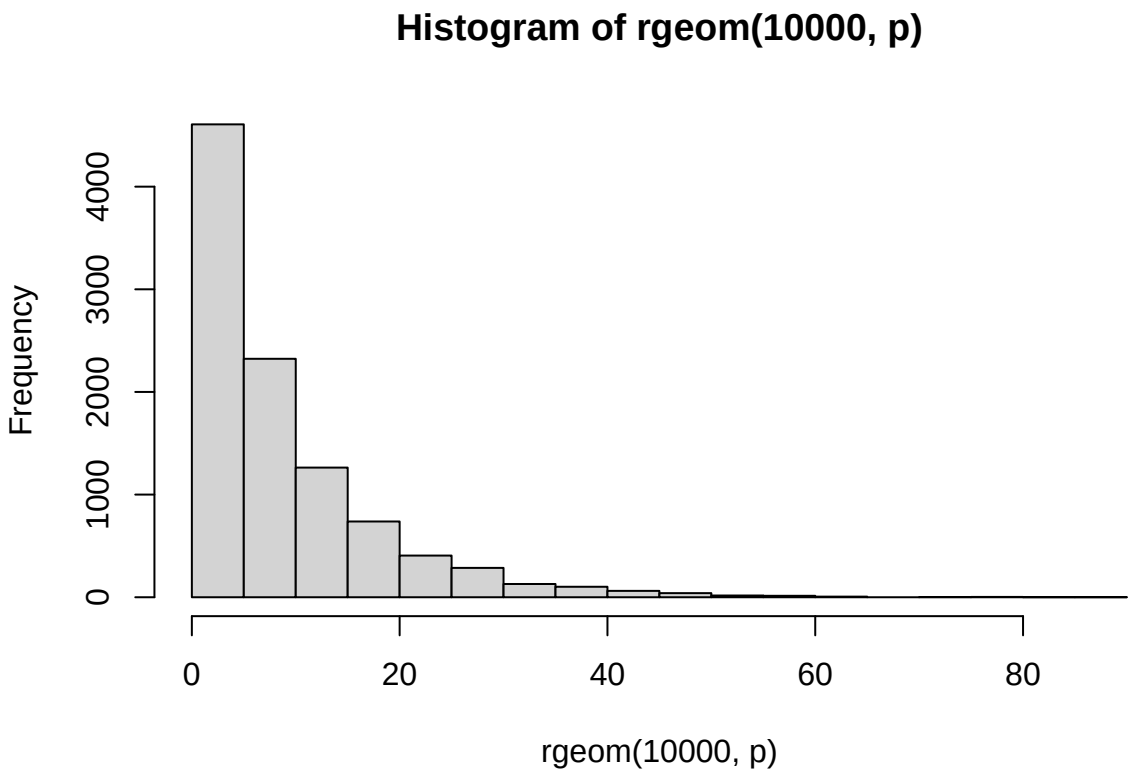

```
qgeom(0.5, p) # El valor que devuelve indica que hay un 50% de que acerte en esa
```

```
## [1] 6
```

```
# cantidad de fracasos o menos antes del exito
qgeom(0.75, p)
```

```
## [1] 13
```

```
hist(rgeom(10000, p))
```



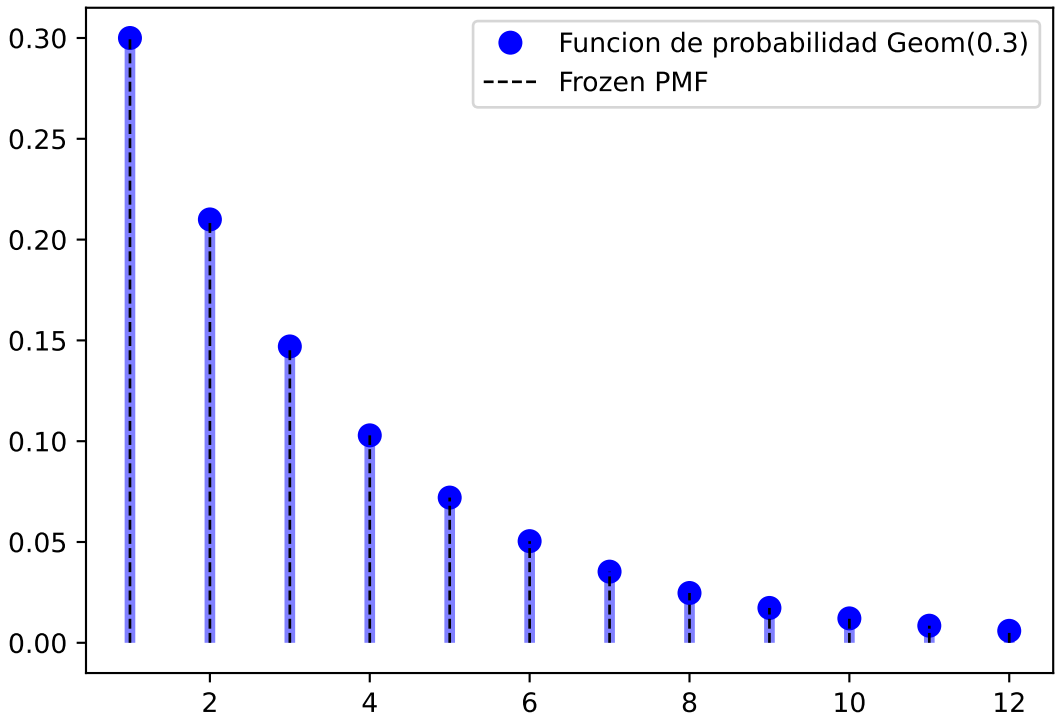
En Python

```
from scipy.stats import geom
import matplotlib.pyplot as plt
import numpy as np

fig, ax = plt.subplots(1,1)
p = 0.3

mean, var, skew, kurt = geom.stats(p, moments = "mvsk")
print(mean, var, skew, kurt)

# Funcion de densidad
```

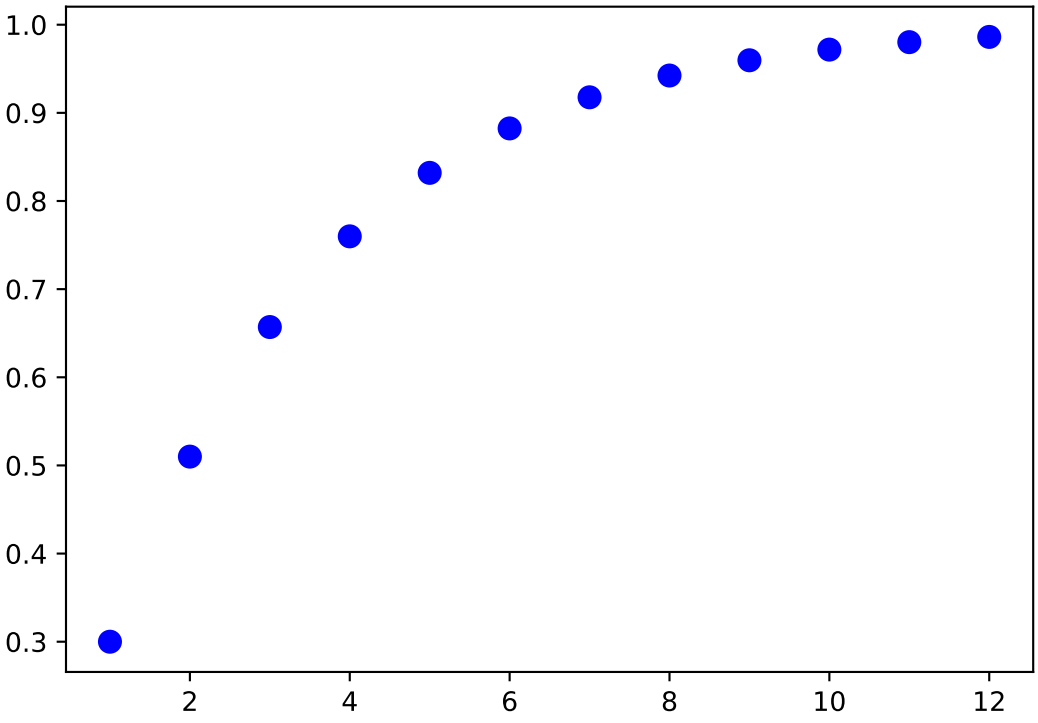
```
## 3.3333333333333335 7.777777777777779 2.031888635868469 6.128571428571429
```

```
x = np.arange(geom.ppf(0.01, p), geom.ppf(0.99, p))
ax.plot(x, geom.pmf(x, p), "bo", ms = 8, label = "Funcion de probabilidad Geom(0.3)")
ax.vlines(x, 0, geom.pmf(x,p), colors = "b", lw = 4, alpha = 0.5)
rv = geom(p)
ax.vlines(x, 0, rv.pmf(x), colors = "k", linestyles = "--", lw = 1, label = "Frozen PMF")
ax.legend(loc = "best")
plt.show()
```

```
# Funcion de distribucion
# El valor que tenga en el eje de las Y al 0.50, equivale al quartil 0.50
```



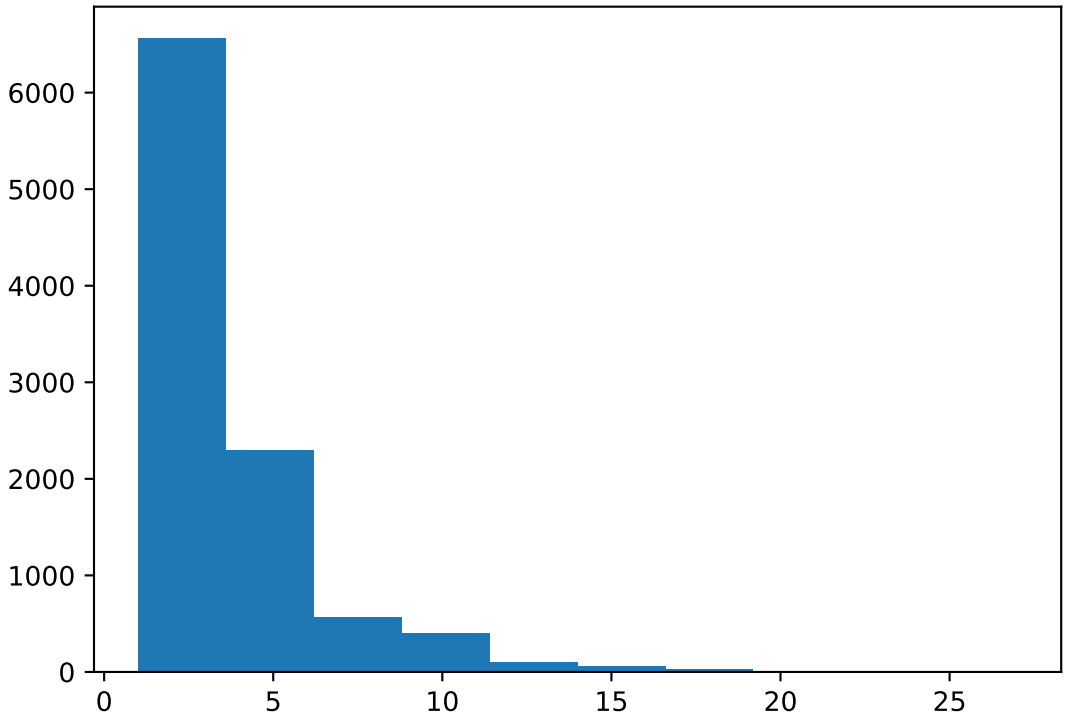

```
fix, ax = plt.subplots(1, 1)
prob = geom.cdf(x, p)
ax.plot(x, prob, "bo", ms = 8, label = "Funcion de distribucion")
plt.show()

# Histograma con numeros random
```




```
fix, ax = plt.subplots(1, 1)
r = geom.rvs(p, size = 10000)
plt.hist(r)
```

```
## (array([6.562e+03, 2.289e+03, 5.660e+02, 3.970e+02, 1.000e+02, 5.500e+01,
##         2.200e+01, 6.000e+00, 0.000e+00, 3.000e+00]), array([ 1. ,  3.6,  6.2,  8.8, 11.4, 14. , 16.6
```

```
plt.show()
```


6000
5000
4000
3000
2000
1000
0
0
5
10
15
20
25