Actividad 1: Simulación Estocástica

Curso: Temas Selectos I: O25 LAT4032 1
Profesor: Rubén Blancas Rivera
Alumnos: auau 1, auau 2, auau 3
Universidad: Universidad de las Américas Puebla
Fecha: 2025-08-15

Tabla de Contenidos

Ejercicio 1

Si $x_0 = 5$ y $x_n = 2x_{n-1}\ mod\ 150$. Encontrar $x_1, \ldots, x_{10}$.

$$x_n = ax_{n-1}\ mod\ m$$

$10 = 2 * 5\ mod\ 15020 = 10 * 5\ mod\ 15040 = 20 * 5\ mod\ 15080 = 40 * 5\ mod\ 15010 = 80 * 5\ mod\ 150\ \vdots$

[1]:
```
pseudoaleatorios = []

x0 = 5
a = 2
m = 150

for i in range(10):
    xn = (a * x0) % m
    x0 = xn
    pseudoaleatorios.append(xn)

print(pseudoaleatorios)
```

```
[10, 20, 40, 80, 10, 20, 40, 80, 10, 20]
```

Ejercicio 3

$$\int_0^1 exp(e^x)\, dx$$

Sea

$$\theta = \int_0^1 exp(e^x)\, dx.$$

Reescritura como valor esperado con $U \sim \text{Unif}(0,1)$:

$$\theta = \mathbb{E}[exp(e^U)]\,.$$

Estimador Monte Carlo con $u_1, \dots, u_K \overset{\text{iid}}{\sim} \text{Unif}(0,1)$:

$$\hat{\theta}_K = \frac{1}{K}\sum_{i=1}^{K} exp(e^{u_i}).$$

[154]:

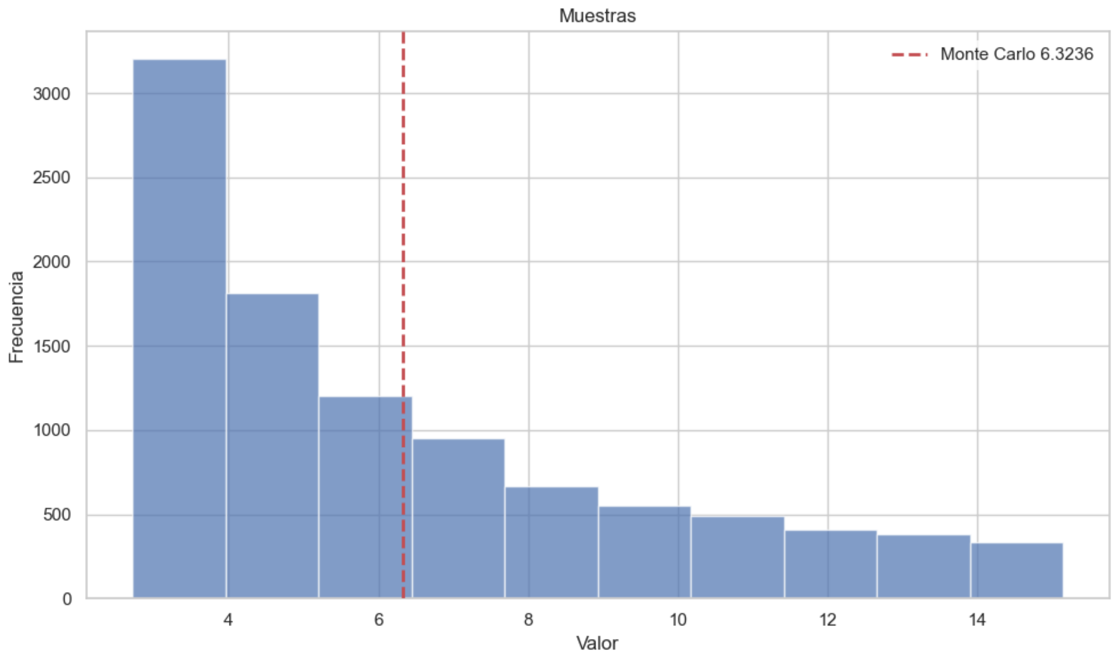
```
import numpy as np

def h(u):
    return np.exp(np.exp(u))

k = 10000

u = np.random.random(k)

muestras = h(u)
montecarlo = muestras.mean()
```

[146]:

```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style='whitegrid')

def histograma(muestras, montecarlo, bins=50):
    fig, ax = plt.subplots(figsize=(10, 6))
    n, _, _ = ax.hist(muestras, bins=bins, alpha=0.7)
    ax.axvline(montecarlo, color='r', linestyle='--',
               linewidth=2, zorder=3, label=f'Monte Carlo {montecarlo:.4f}')
    xmin, xmax = ax.get_xlim()
    if montecarlo < xmin or montecarlo > xmax:
        ax.set_xlim(min(xmin, montecarlo), max(xmax, montecarlo))
    ax.set_title('Muestras')
    ax.set_xlabel('Valor')
    ax.set_ylabel('Frecuencia')
    ax.legend(facecolor='white', edgecolor='none')
    plt.tight_layout()
```

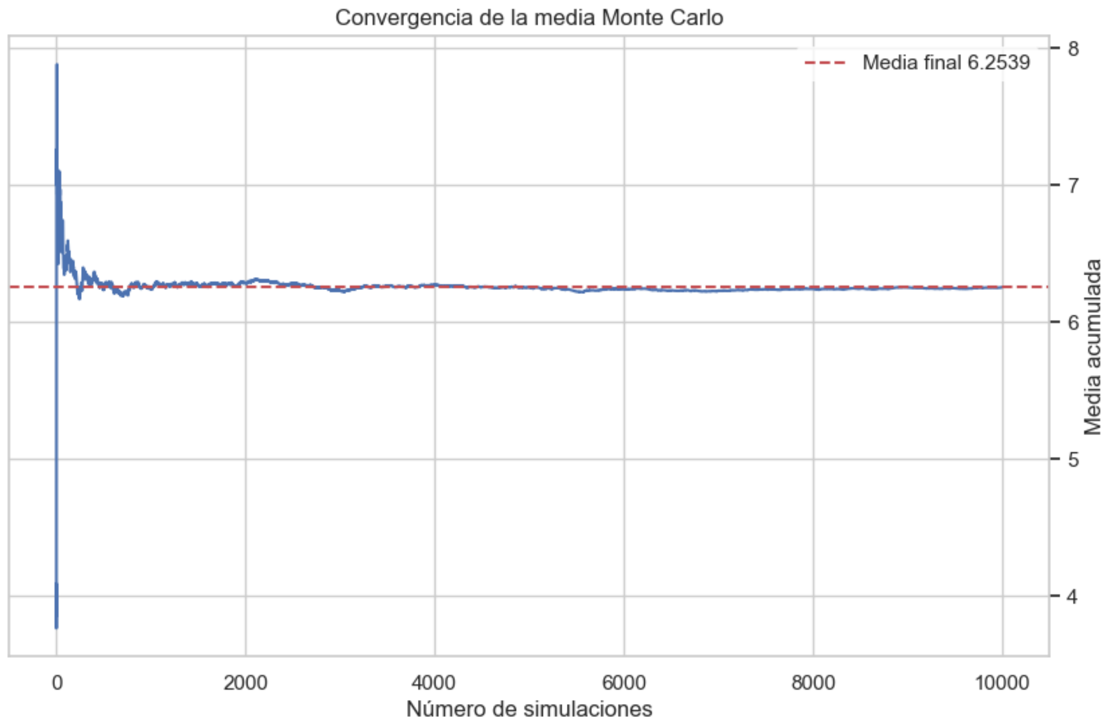
```
    plt.show()
```

[155]:
```
histograma(muestras, montecarlo, bins=10)
```



[ ]:
```
def tlc(muestras, valor_verdadero=None):

    k = len(muestras)
    medias = np.cumsum(muestras) / np.arange(1, k + 1)

    plt.figure(figsize=(10, 6))
    plt.plot(medias)
    plt.xlabel('Número de simulaciones')
    plt.ylabel('Media acumulada')
    plt.title(f'Convergencia de la media Monte Carlo')
    plt.axhline(y=medias[-1], color='r', linestyle='--', label=f'Media final␣
↪{medias[-1]:.4f}')
    if valor_verdadero is not None:
        plt.axhline(y=valor_verdadero, color='g', linestyle='--', label=f'Valor␣
↪verdadero {valor_verdadero:.4f}')
    plt.legend(facecolor='white', edgecolor='none')
    plt.gca().yaxis.set_ticks_position('right')
    plt.gca().yaxis.set_label_position('right')
    plt.show()
```

[95]:
```
tlc(muestras)
```


Convergencia de la media Monte Carlo
Media final 6.2539
Media acumulada
Número de simulaciones
0
2000
4000
6000
8000
10000
4
5
6
7
8

Ejercicio 5

$$\int_{-2}^{2} e^{x+x^2}\,dx$$

Sea

$$\theta = \int_{-2}^{2} e^{x+x^2}\,dx.$$

Cambio de variable a ([0,1]):

$$u = \frac{x-(-2)}{2-(-2)} = \frac{x+2}{4}, \qquad x = -2+4u, \qquad dx = 4\,du.$$

Entonces

$$\theta = \int_{0}^{1} 4\ exp\big[(-2+4u) + (-2+4u)^2\big]\,du.$$

Forma de valor esperado con $U \sim \mathrm{Unif}(0,1)$:

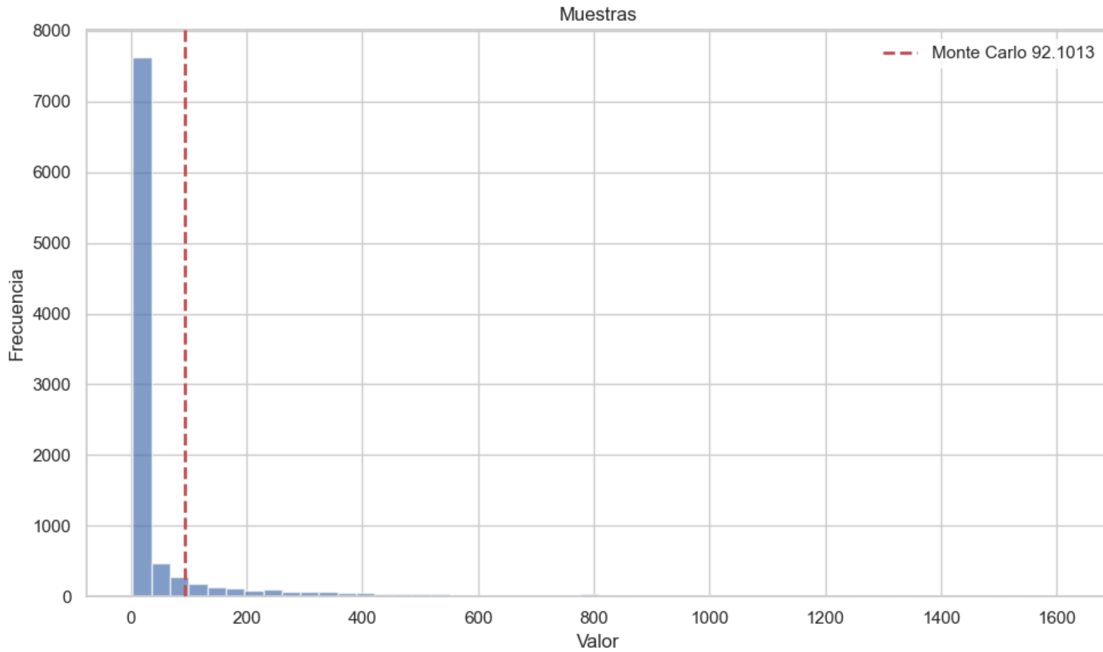
$$\theta = \mathbb{E}[g(U)]\,, \qquad g(u) = 4\ exp\big[(-2+4u) + (-2+4u)^2\big].$$

Estimador Monte Carlo:

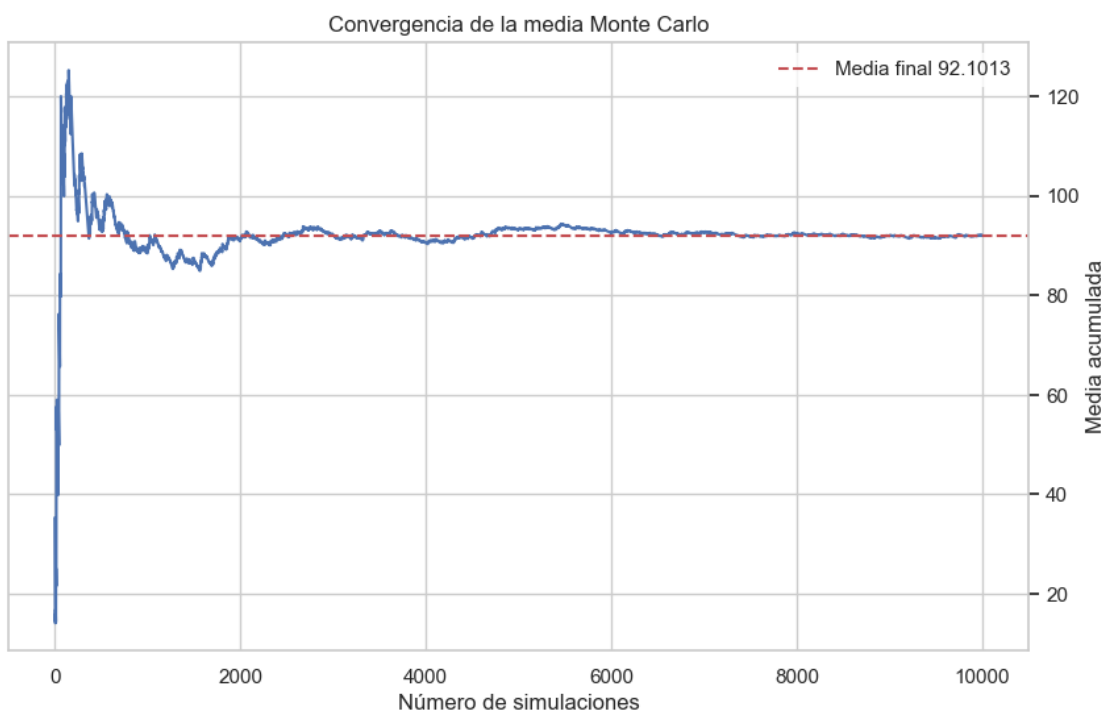
$$\hat{\theta}_K = \frac{1}{K}\sum_{i=1}^{K} g(u_i), \qquad u_i \overset{iid}{\sim} \mathrm{Unif}(0,1).$$

[ ]:
```
def h(u):
    return (b-a)*np.exp(a+(b-a)*u + (a+(b-a)*u)**2)

k = 10000
a = -2
b = 2
u = np.random.random(k)

muestras = h(u)
montecarlo = muestras.mean()
```

[89]:
```
histograma(muestras, montecarlo)
```



[90]:
```
tlc(muestras)
```



Ejercicio 7

$$\int_0^\infty \frac{x}{(1+x^2)^2}\,dx$$

Sea

$$\theta = \int_0^\infty \frac{x}{(1+x^2)^2}\,dx.$$

Cambio:

$$y = \frac{1}{x+1}, \qquad dy = -\frac{dx}{(x+1)^2} = -y^2\,dx.$$

Entonces

$$\theta = \int_0^1 h(y)\,dy, \qquad h(y) = \frac{g\left(\frac{1}{y}-1\right)}{y^2}, \quad g(x) = \frac{x}{(1+x^2)^2}.$$

Cálculo explícito de h:

$$x = \frac{1-y}{y} \;\Rightarrow\; h(y) = \frac{(1-y)\,y}{\left(1-2y+2y^2\right)^2}, \qquad y \in (0,1).$$

Forma de esperanza con $U \sim \text{Unif}(0,1)$:

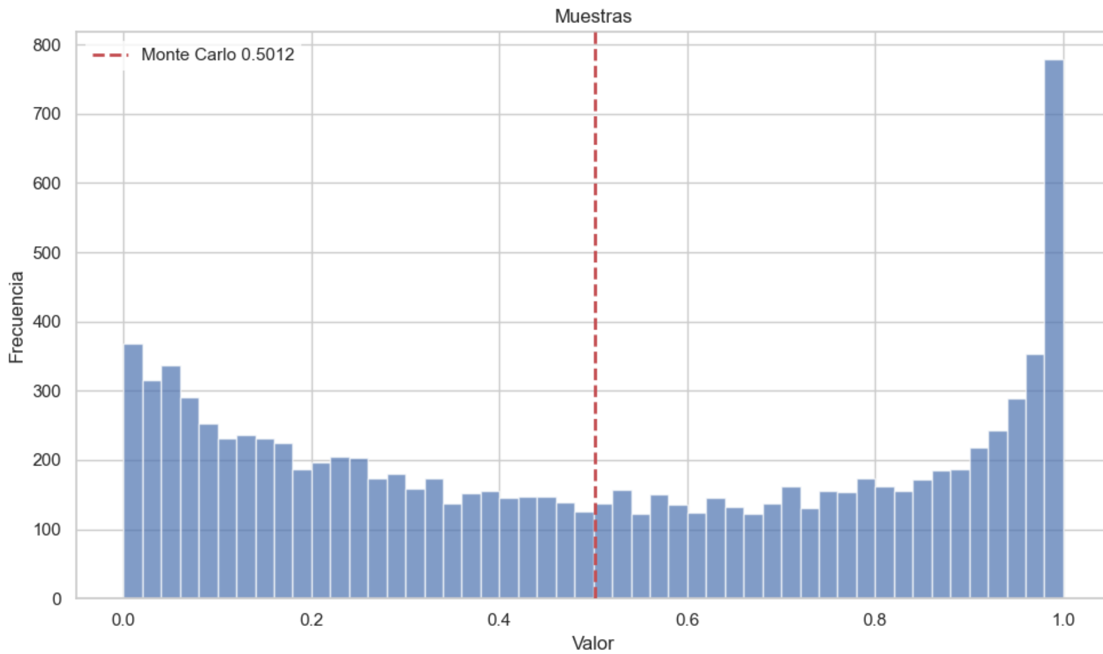
$$\theta = \mathbb{E}[\,h(U)\,].$$

Estimador Monte Carlo:

$$\hat{\theta}_K = \frac{1}{K}\sum_{i=1}^{K} h(u_i), \quad u_i \overset{\text{iid}}{\sim} \text{Unif}(0,1).$$

[99]:
```
def h(u):
    return ((((1/u)-1)/(1+((1/u)-1)**2)**2))/(u**2)

k = 10_000

u = np.random.random(k)
muestras = h(u)
montecarlo = muestras.mean()
```

[102]:
```
histograma(muestras, montecarlo)
```



Sea

$$\theta = \int_0^\infty \frac{x}{(1+x^2)^2}\,dx.$$

Integral impropia:

$$\theta = \lim_{b\to\infty} \int_0^b \frac{x}{(1+x^2)^2}\,dx.$$

Sustitución $u = 1 + x^2 \Rightarrow du = 2x\,dx$: cuando $x = 0 \Rightarrow u = 1$, cuando $x = b \Rightarrow u = 1 + b^2$. Entonces

$$\int_0^b \frac{x}{(1+x^2)^2}\,dx = \frac{1}{2}\int_1^{1+b^2} u^{-2}\,du.$$

Primitiva:

$$\int u^{-2}\,du = -u^{-1} + C.$$

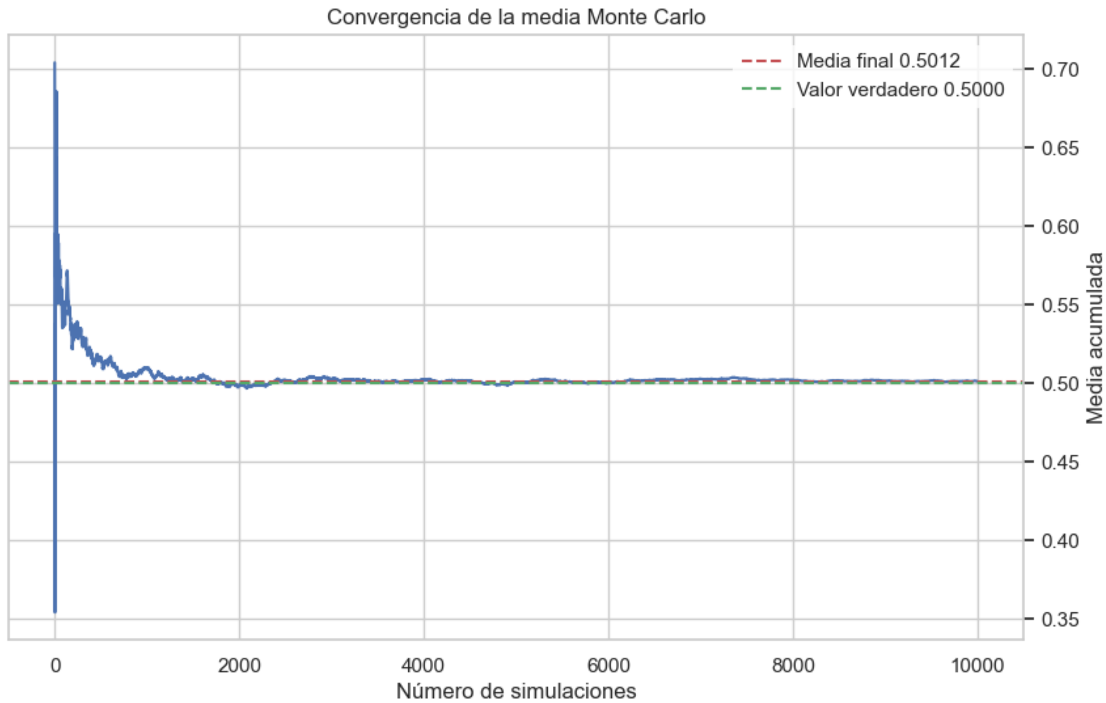
Evaluación:

$$\frac{1}{2}\Big[-u^{-1}\Big]_1^{1+b^2} = \frac{1}{2}\Big(-\frac{1}{1+b^2} + 1\Big).$$

Límite:

$$\theta = \lim_{b\to\infty} \frac{1}{2}\Big(1 - \frac{1}{1+b^2}\Big) = \frac{1}{2}.$$

[101]:
```
tlc(muestras, 1/2)
```


Convergencia de la media Monte Carlo
Media final 0.5012
Valor verdadero 0.5000
0.70
0.65
0.60
0.55
0.50
0.45
0.40
0.35
Media acumulada
0
2000
4000
6000
8000
10000
Número de simulaciones

Ejercicio 9

$$\int_0^1 \int_0^1 e^{(x+y)^2}\, dy\, dx$$

Sea

$$\theta = \int_0^1 \int_0^1 e^{(x+y)^2}\, dy\, dx.$$

Entonces

$$\theta = \int_0^1 \int_0^1 g(x_1, x_2)\, dx_1\, dx_2, \qquad g(x_1, x_2) = e^{(x_1+x_2)^2}.$$

Sabemos que:

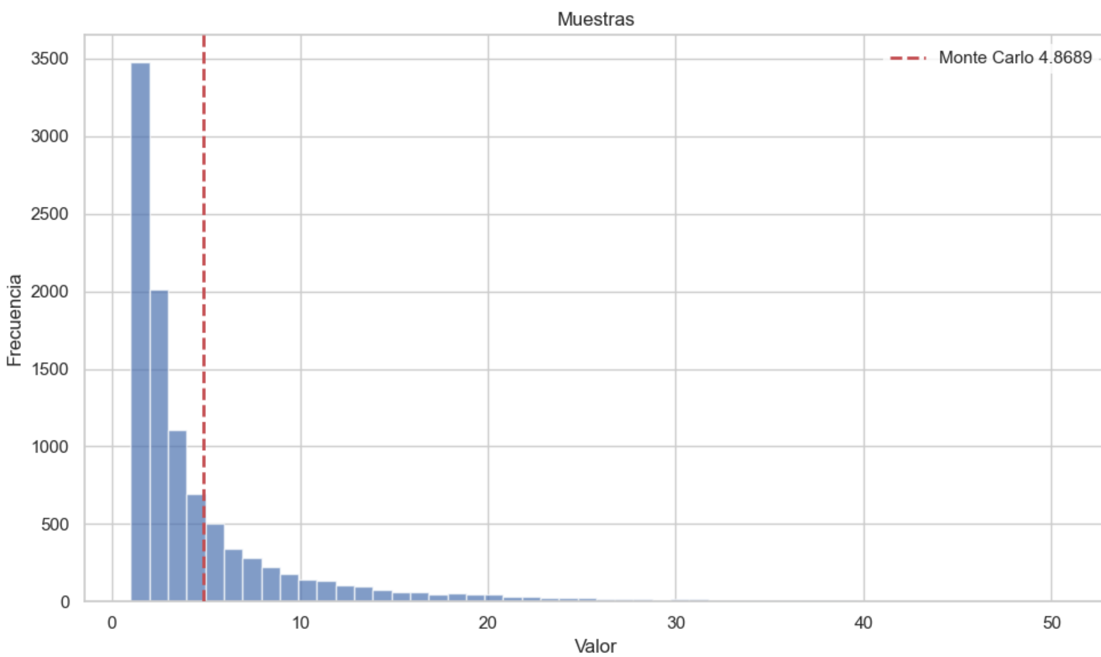
$$\theta = \mathbb{E}[g(U_1, U_2)], \quad U_1, U_2 \overset{iid}{\sim} \text{Unif}(0, 1).$$

Estimador Monte Carlo:

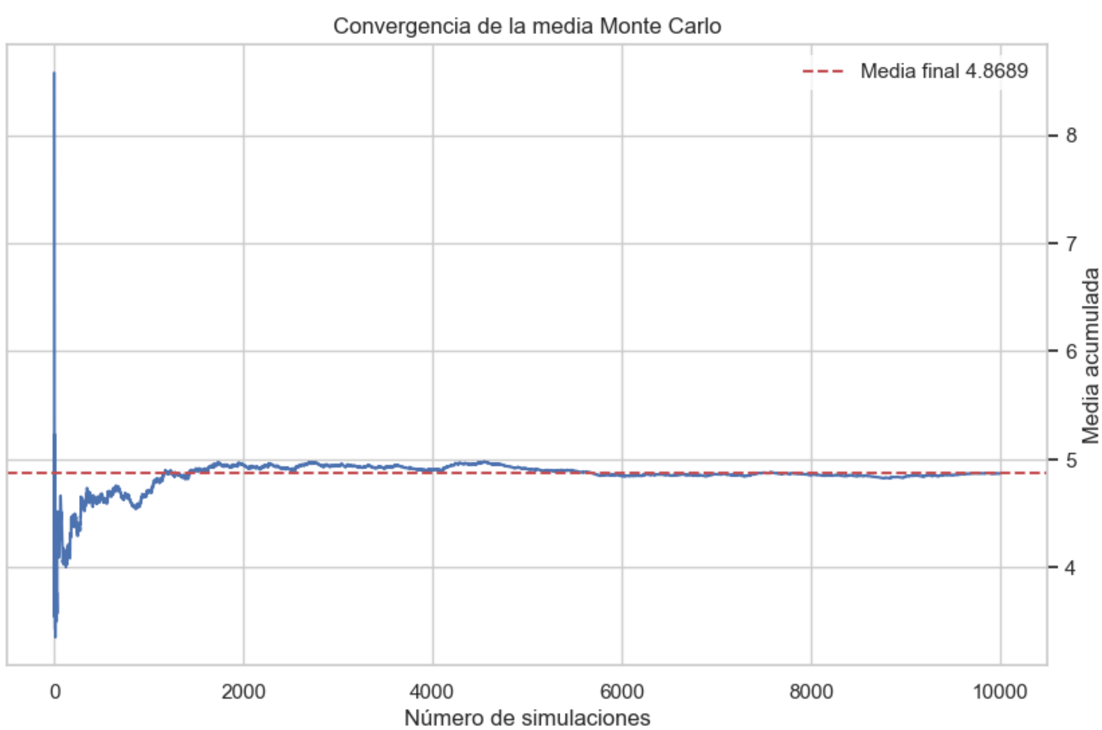
$$\hat{\theta}_k = \frac{1}{k} \sum_{i=1}^{k} g(u_{i1}, u_{i2}) = \frac{1}{k} \sum_{i=1}^{k} exp((u_{i1} + u_{i2})^2), \quad (u_{i1}, u_{i2}) \overset{iid}{\sim} \text{Unif}(0, 1).$$

[103]:
```
def h(u1, u2):
    return np.exp((u1 + u2)**2)

k = 10000

u1 = np.random.random(k)
u2 = np.random.random(k)
muestras = h(u1, u2)
montecarlo = muestras.mean()
```

[104]:
```
histograma(muestras, montecarlo)
```



[105]:
```
tlc(muestras)
```



Ejercicio

Usar simulación para aproximar $Cov(U, e^U)$, donde $U \sim \mathcal{U}(0,1)$. Comparar con la respuesta exacta.

Asumo $U \sim$ Unif$(0,1)$.

Por definición,

$$Cov(X,Y) = \mathbb{E}[(X - \mathbb{E}X)(Y - \mathbb{E}Y)].$$

Expansión lineal:

$$Cov(X,Y) = \mathbb{E}[XY] - \mathbb{E}[X]\,\mathbb{E}[Y].$$

Aplicando a $X = U$ y $Y = e^U$:

$$Cov(U,e^U) = \mathbb{E}[Ue^U] - \mathbb{E}[U]\,\mathbb{E}[e^U].$$

Cálculo analítico

$$\mathbb{E}[U] = \int_0^1 u\,du = \frac{1}{2}.$$

$$\mathbb{E}[e^U] = \int_0^1 e^u\,du = e - 1.$$

$$\mathbb{E}[Ue^U] = \int_0^1 ue^u\,du = [ue^u]_0^1 - \int_0^1 e^u\,du = e - (e-1) = 1.$$

Por tanto,

$$Cov(U,e^U) = 1 - \tfrac{1}{2}(e-1) \;=\; 0.140859086$$

Estimación Monte Carlo

Sea $u_1, \dots, u_K \overset{iid}{\sim}$ Unif$(0,1)$. Entonces

$$\hat{\mu}_U = \frac{1}{K}\sum_{i=1}^{K} u_i, \qquad \hat{\mu}_e = \frac{1}{K}\sum_{i=1}^{K} e^{u_i}, \qquad \widehat{m} = \frac{1}{K}\sum_{i=1}^{K} u_i e^{u_i}.$$

Estimador por identidad de momentos:

$$\widehat{Cov}^{(MC)} = \widehat{m} - \hat{\mu}_U\,\hat{\mu}_e$$

[119]:
```
def valor_esperado_1(u):
    return u * np.exp(u)
```

```
def valor_esperado_2(u):
    return u

def valor_esperado_3(u):
    return np.exp(u)

k = 1000000

u = np.random.random(k)

montecarlo = valor_esperado_1(u).mean() - valor_esperado_2(u).mean() *␣
 ↪valor_esperado_3(u).mean()
```

[120]:
```
montecarlo
```

[120]: np.float64(0.1406986786630252)

[121]:
```
1 - 1/2*(np.e -1)
```

[121]: 0.14085908577047745

Ejercicio

Para variables aleatorias uniformes $U_1, U_2, ...$ definir

$$N = min\left\{n : \sum_{i=1}^{n} U_i > 1\right\}.$$

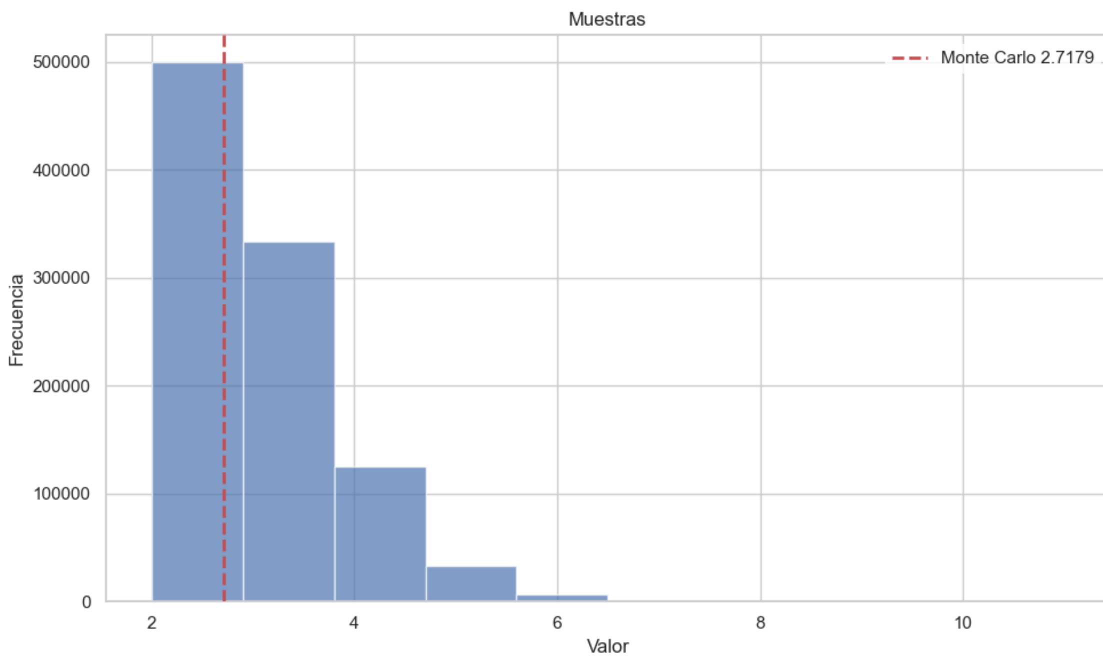
Estimar $\mathbb{E}[N]$ por simulación con: a) 100 valores, b) 1000 valores, c) 10000 valores, d) Discutir el valor esperado.

```
PSEUDOCÓDIGO - MINIMO_N(k)
total_contadores ← 0
PARA i ← 1 HASTA k HACER:
    suma ← 0
    contador ← 0
    MIENTRAS suma < 1 HACER:
        contador ← contador + 1
        suma ← UNIFORME(0,1)
    total_contadores ← total_contadores + contador
RETORNAR total_contadores / k
```

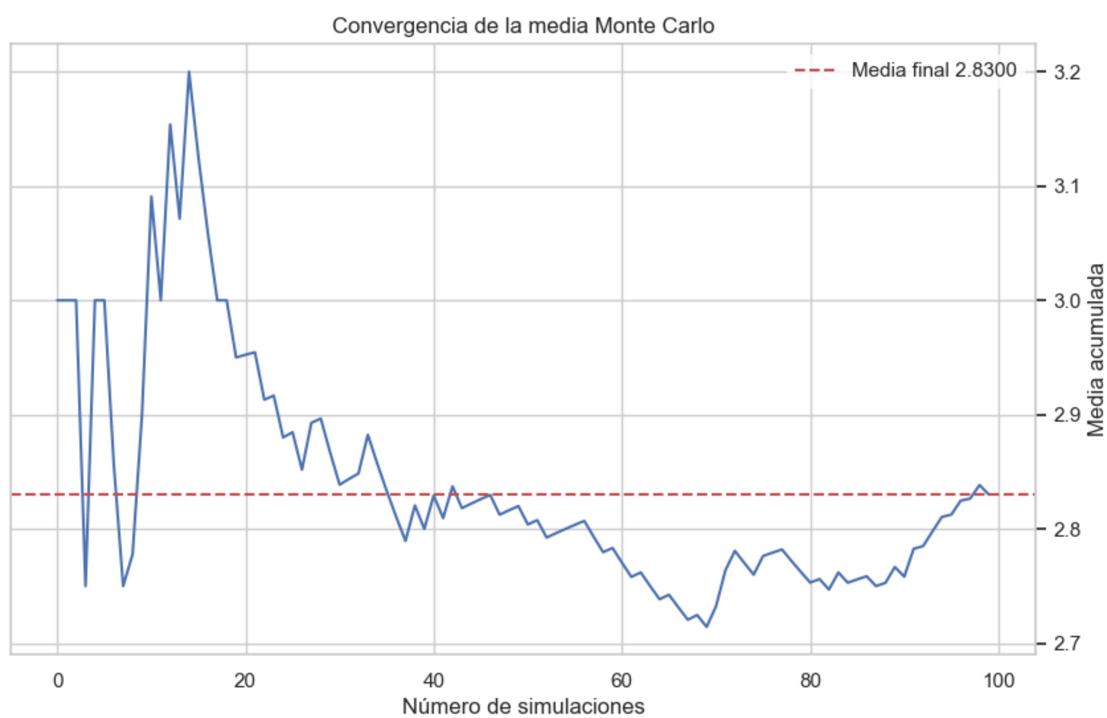
[123]:
```python
def minimo_N(k):
    lista_contadores = []
    for _ in range(k):
        suma = 0
        contador = 0
        while suma < 1:
            contador += 1
            suma += np.random.random()
        lista_contadores.append(contador)
    return lista_contadores
```

[142]:
```python
muestras = minimo_N(1000000)
montecarlo = np.mean(muestras)
```

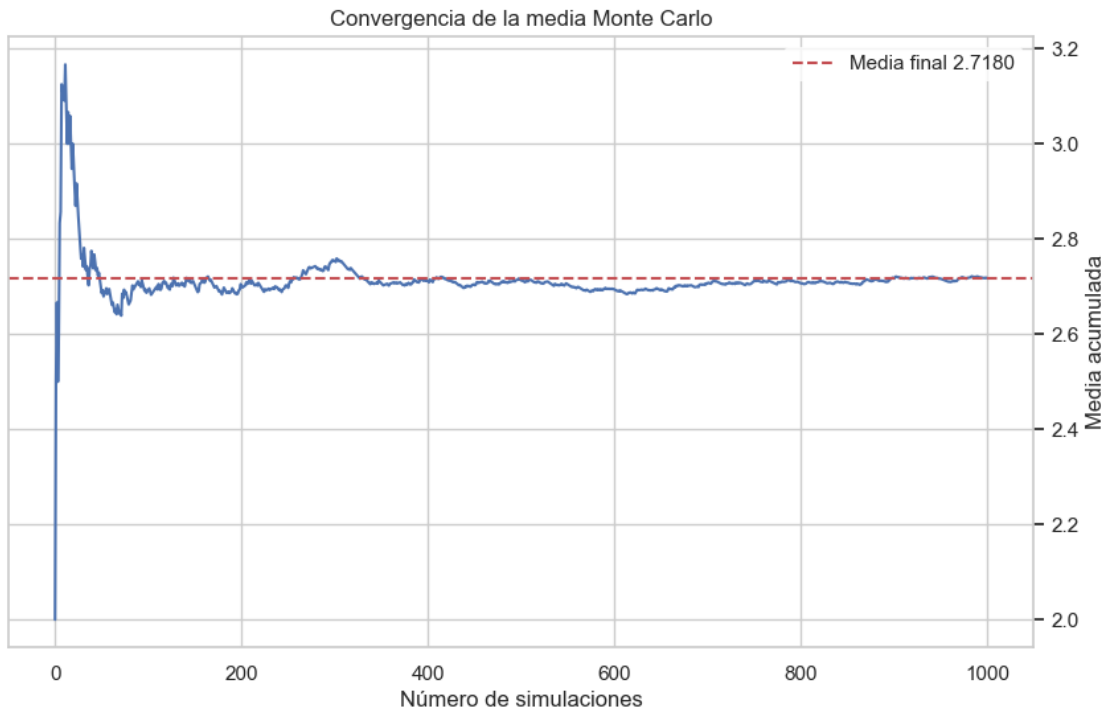
[151]:
```python
histograma(muestras, montecarlo, bins=10)
```



[138]:
```
tlc(minimo_N(100))
```



[139]:
```
tlc(minimo_N(1000))
```



[140]:
```
tlc(minimo_N(10000))
```