Laboratorio 1 - Modelación y Simulación

Roberto Vallecillos Carol Arevalo

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as stats
from scipy.stats import norm, chisquare, expon, uniform
from scipy.stats import skew, kurtosis
```

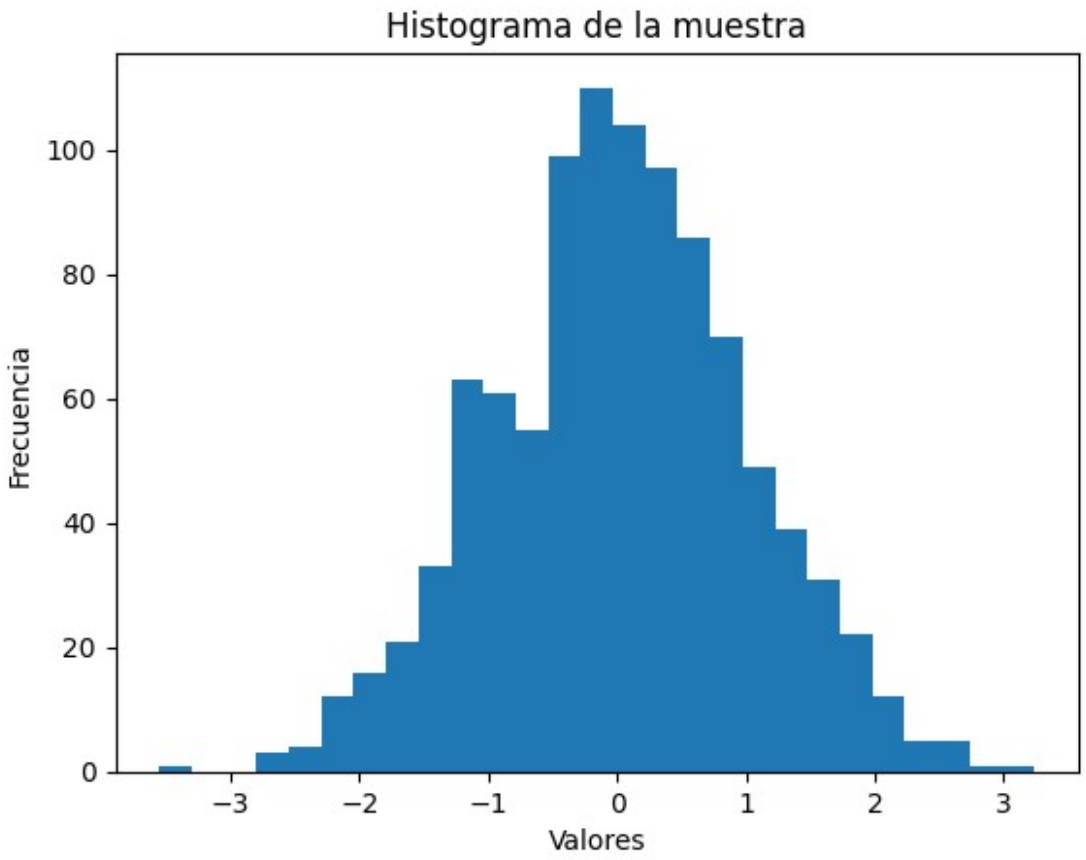
Ejercicio 1

Genere una muestra aleatoria de 1000 puntos de datos de una distribución normal con media 0 y desviación estándar 1. Tasks:

1. Calcule e imprima la media, la varianza y la asimetría de la muestra.
2. Trace un histograma de los datos y observe su forma.
3. ¿Qué le dicen estas estadísticas sobre la tendencia central, la dispersión y la forma de los datos?

```
# Generar una muestra aleatoria de 1000 puntos de datos de una
distribución normal con media 0 y desviación estándar 1
muestra = np.random.normal(0, 1, size=1000)

# Calcular la media de la muestra
media = np.mean(muestra)

# Calcular la varianza de la muestra
varianza = np.var(muestra)

# Calcular la asimetría de la muestra
asimetria = skew(muestra)

# Imprimir la media, la varianza y la asimetría
print("Media de la muestra:", media)
print("Varianza de la muestra:", varianza)
print("Asimetría de la muestra:", asimetria)

# Trazar un histograma de los datos
plt.hist(muestra, bins='auto')
plt.xlabel("Valores")
plt.ylabel("Frecuencia")
plt.title("Histograma de la muestra")
plt.show()
```

```
Media de la muestra: 0.024024902157106325
Varianza de la muestra: 0.9820105726440311
Asimetría de la muestra: -0.0005972564507581143
```



- Tendencia central: Como la media es cero, se puede ver que los datos tienden a agruparse alrededor de ese valor central.
- Dispersión: Como la varianza es 1, los datos se dispersan de manera moderada alrededor de la media.
- Forma: Como la asimetría está cerca de cero, los datos tienen una distribución simétrica en forma de campana, típica de una distribución normal.

En resumen, a través e estas estadísticas se puede comprobar que la distribución de los datos es normal.

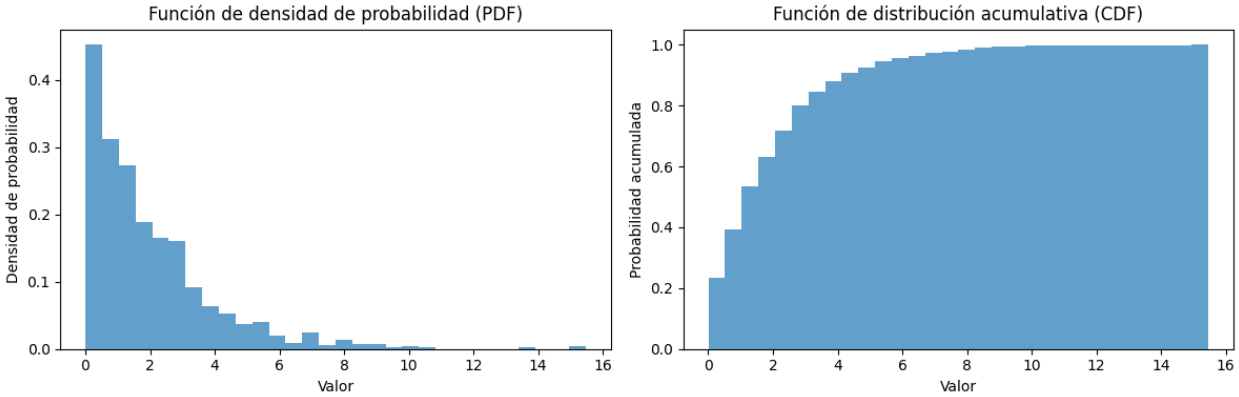
Ejercicio 2

Genere una muestra aleatoria de 1000 puntos de datos a partir de una distribución exponencial con parámetro de tasa 0.5. Tasks:

1. Trace la función de densidad de probabilidad (PDF) y la función de distribución acumulativa (CDF) de los datos.
2. Calcule e imprima las probabilidades para intervalos específicos utilizando la CDF. (por lo menos 2 intervalos a su elección)
3. ¿Qué información puede obtener de la forma del PDF y el comportamiento de la CDF?

```
# Generar 1000 puntos de datos a partir de una distribución
exponencial con tasa 0.5
```

```
data = np.random.exponential(scale=1/0.5, size=1000)

# 1. Trazar la función de densidad de probabilidad (PDF) y la función
de distribución acumulativa (CDF)
plt.figure(figsize=(12, 4))

# PDF
plt.subplot(1, 2, 1)
plt.hist(data, bins=30, density=True, alpha=0.7)
plt.xlabel('Valor')
plt.ylabel('Densidad de probabilidad')
plt.title('Función de densidad de probabilidad (PDF)')

# CDF
plt.subplot(1, 2, 2)
plt.hist(data, bins=30, density=True, cumulative=True, alpha=0.7)
plt.xlabel('Valor')
plt.ylabel('Probabilidad acumulada')
plt.title('Función de distribución acumulativa (CDF)')

plt.tight_layout()
plt.show()

# 2. Calcular e imprimir las probabilidades para intervalos
específicos utilizando la CDF
intervalo_1 = (0, 2)
intervalo_2 = (2, 5)

probabilidad_intervalo_1 = np.sum((data >= intervalo_1[0]) & (data <
intervalo_1[1])) / len(data)
probabilidad_intervalo_2 = np.sum((data >= intervalo_2[0]) & (data <
intervalo_2[1])) / len(data)

print('Probabilidad para el intervalo', intervalo_1, ':',
probabilidad_intervalo_1)
print('Probabilidad para el intervalo', intervalo_2, ':',
probabilidad_intervalo_2)
```



```
Probabilidad para el intervalo (0, 2) : 0.622
Probabilidad para el intervalo (2, 5) : 0.298
```

La información que se puede obtener de la forma del PDF es cómo se distribuyen los datos en términos de su densidad de probabilidad. En este caso, al tratarse de una distribución exponencial, la forma del PDF será decreciente, lo que indica que los valores más pequeños tienen una mayor densidad de probabilidad que los valores más grandes.

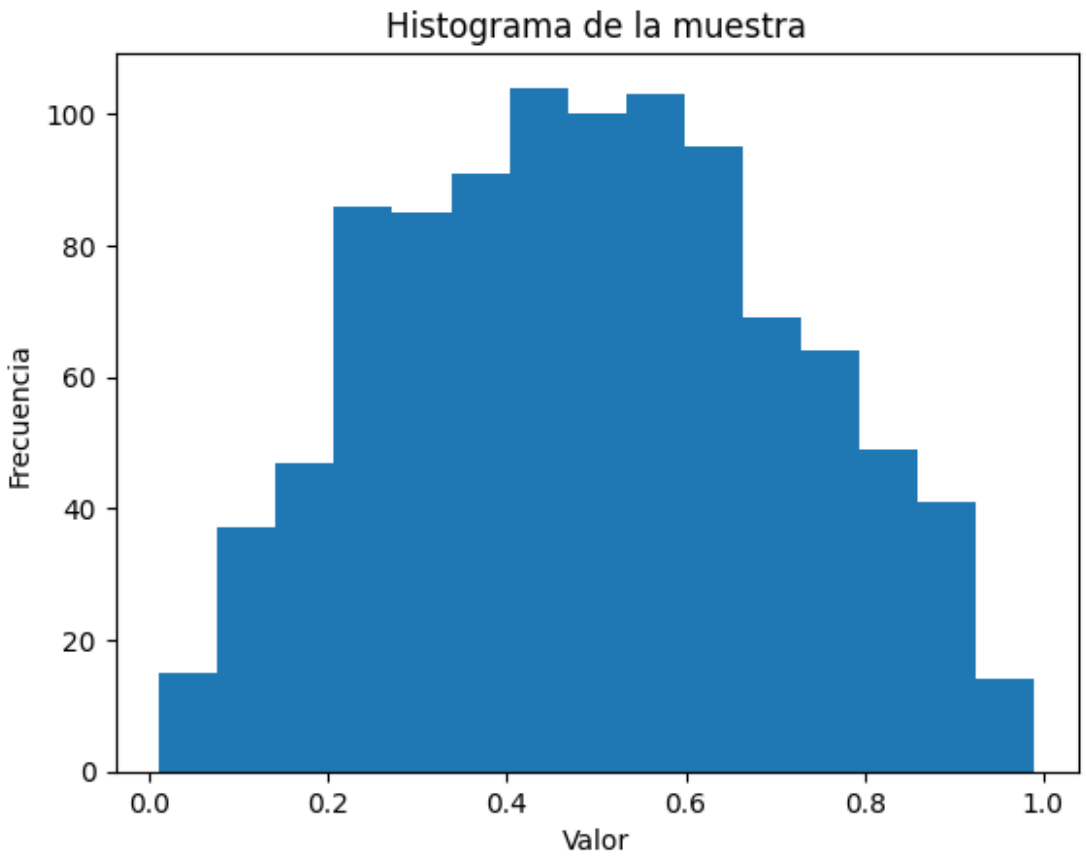
El comportamiento de la CDF muestra cómo se acumula la probabilidad a medida que aumentamos los valores. En este caso, al tratarse de una distribución exponencial, la CDF aumentará gradualmente y se acercará a 1 a medida que los valores aumenten. Esto indica que la probabilidad acumulada de obtener valores más pequeños disminuye a medida que aumentamos los valores.

Ejercicio 3

Genere una muestra aleatoria de 1000 puntos de datos a partir de una distribución beta con parámetros de forma(2, 2). Tasks:

1. Calcule e imprima el sesgo y la curtosis de la muestra.
2. Analice las implicaciones del sesgo y la curtosis en la forma y las características de la distribución.

```
#Genere una muestra aleatoria de 1000 puntos de datos a partir de una distribución beta con parámetros de forma(2, 2).
muestra = np.random.beta(2, 2, 1000)

#mostrar grafica
plt.hist(muestra, bins='auto')
plt.title("Histograma de la muestra")
plt.xlabel("Valor")
plt.ylabel("Frecuencia")
plt.show()

#Calcule e imprima el sesgo y la curtosis de la muestra.
print("Sesgo: ", skew(muestra))
print("Curtosis: ", kurtosis(muestra))
```




```
Sesgo:  0.06593276577040154
Curtosis:  -0.7812178580258564
```

- Sesgo: Un sesgo cercano a 0 significa que la distribución es casi simétrica
- Curtosis: Una curtosis cercana a -1 indica una distribución más aplanada en los extremos _____

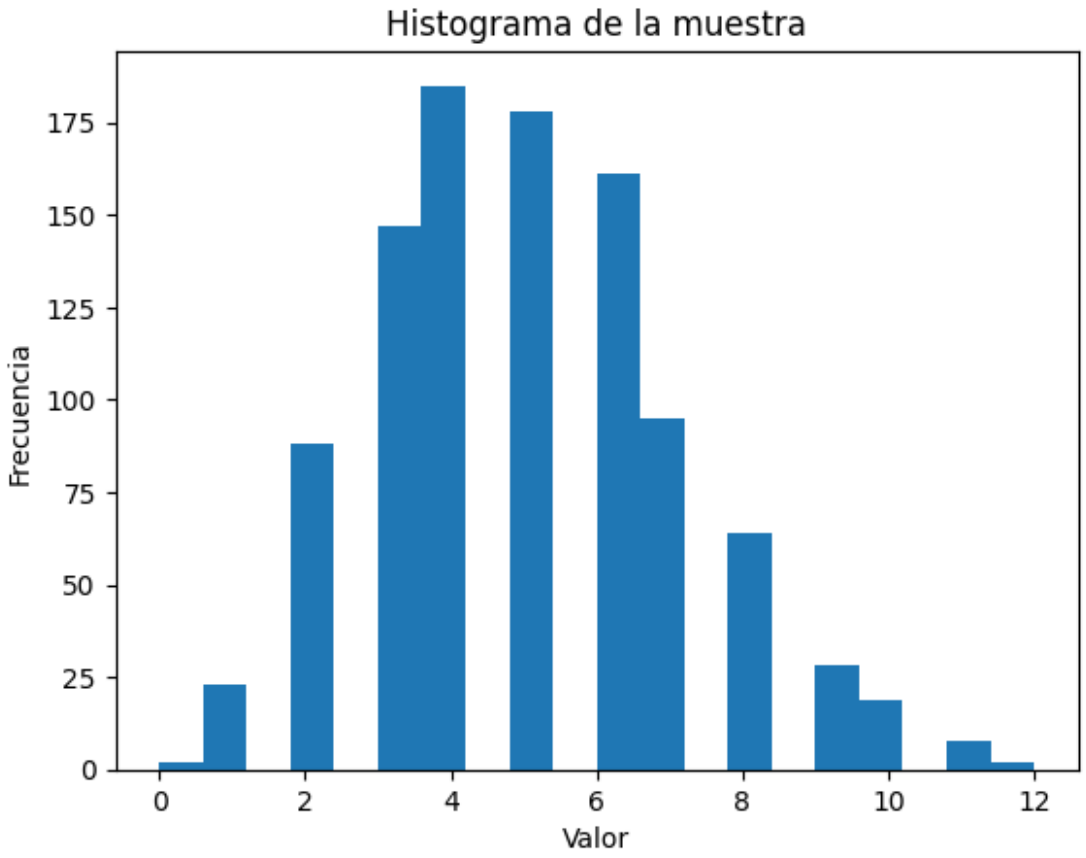
Ejercicio 4

Genere una muestra aleatoria de 1000 puntos de datos a partir de una distribución de Poisson con parámetro lambda 5. Tasks:

1. Trace un histograma de los datos generados.
2. Calcule e imprima la media y la varianza de la muestra.
3. ¿Puedes observar algún patrón o tendencia en los datos generados a partir de la distribución de Poisson?

```
# Generar muestra aleatoria de 1000 puntos de datos
muestra = np.random.poisson(lam=5, size=1000)

# Trace un histograma de los datos generados
plt.hist(muestra, bins='auto')
```

```
plt.xlabel('Valor')
plt.ylabel('Frecuencia')
plt.title('Histograma de la muestra')
plt.show()

# Calcule e imprima la media y la varianza de la muestra
media = np.mean(muestra)
varianza = np.var(muestra)
print('Media:', media)
print('Varianza:', varianza)
```




```
Media: 4.967
Varianza: 4.4079109999999995
```

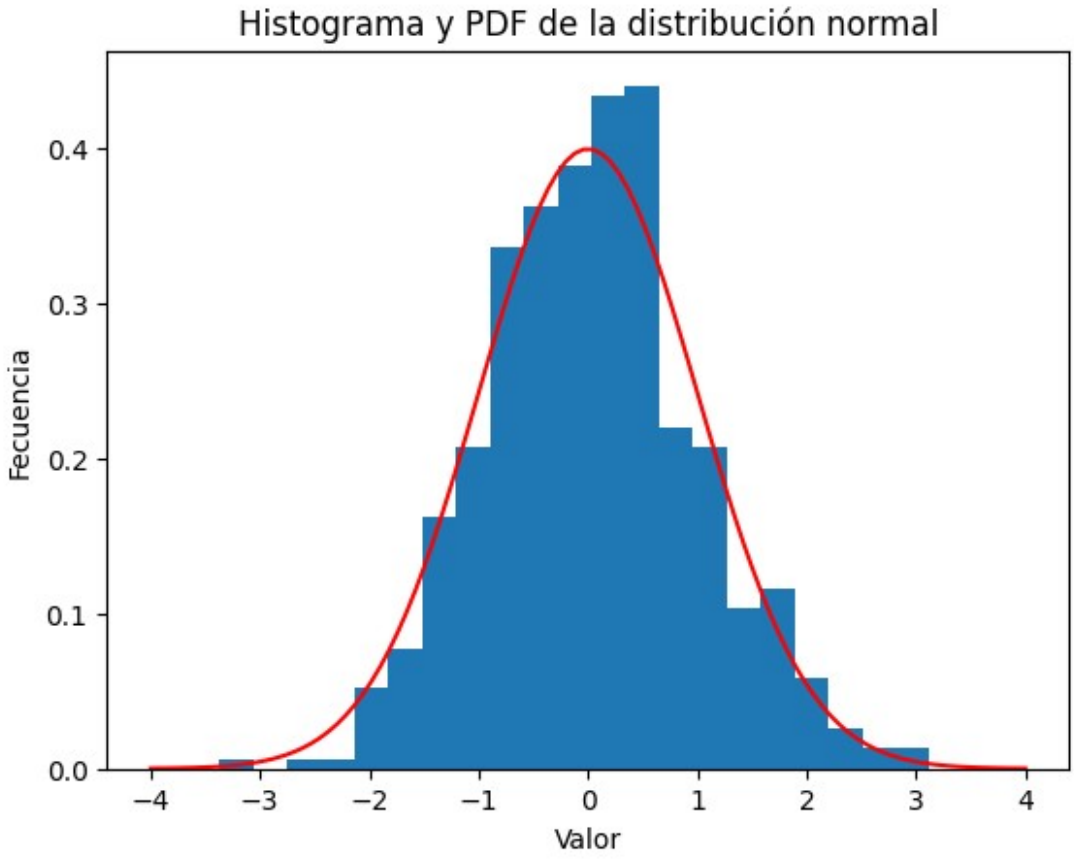
El patrón general que se puede observar en la gráfica es que los valores son más probables alrededor del parámetro lambda y disminuyen a medida que nos alejamos de este valor. Asimismo se puede observar que el valor de lamda es casi igual a la media y varianza de los datos. _____

Ejercicio 5 - Muestreo Muestree 500 puntos de datos de una distribución normal utilizando el método de transformación inversa. Tasks:

1. Trace un histograma de los datos generados.

2. Compare el histograma con la PDF teórica de la distribución normal.
3. ¿La muestra generada representa con precisión la distribución normal subyacente?

```
mu = 0
sigma = 1

uniform_samples = np.random.uniform(size=500)
normal_samples = norm.ppf(uniform_samples)

pdf_x = np.linspace(mu - 4*sigma, mu + 4*sigma, 100)
pdf_y = 1/(sigma * np.sqrt(2*np.pi)) * np.exp(-(pdf_x - mu)**2 /
(2*sigma**2))
plt.plot(pdf_x, pdf_y, 'r', label='PDF teórica')

plt.hist(normal_samples, bins='auto', density=True, label='PDF
muestral')
plt.xlabel('Valor')
plt.ylabel('Fecuencia')
plt.title('Histograma y PDF de la distribución normal')
plt.show()
```



- Como se puede observar, los datos generados se asemejan al comportamiento esperado de una distribución normal y siguen la forma de la PDF teórica. Aunque no la representa con precisión, en general si tiene la misma forma y comportamiento: hay menos datos en

los extremos y más datos cerca de cero, formando una forma de campana.

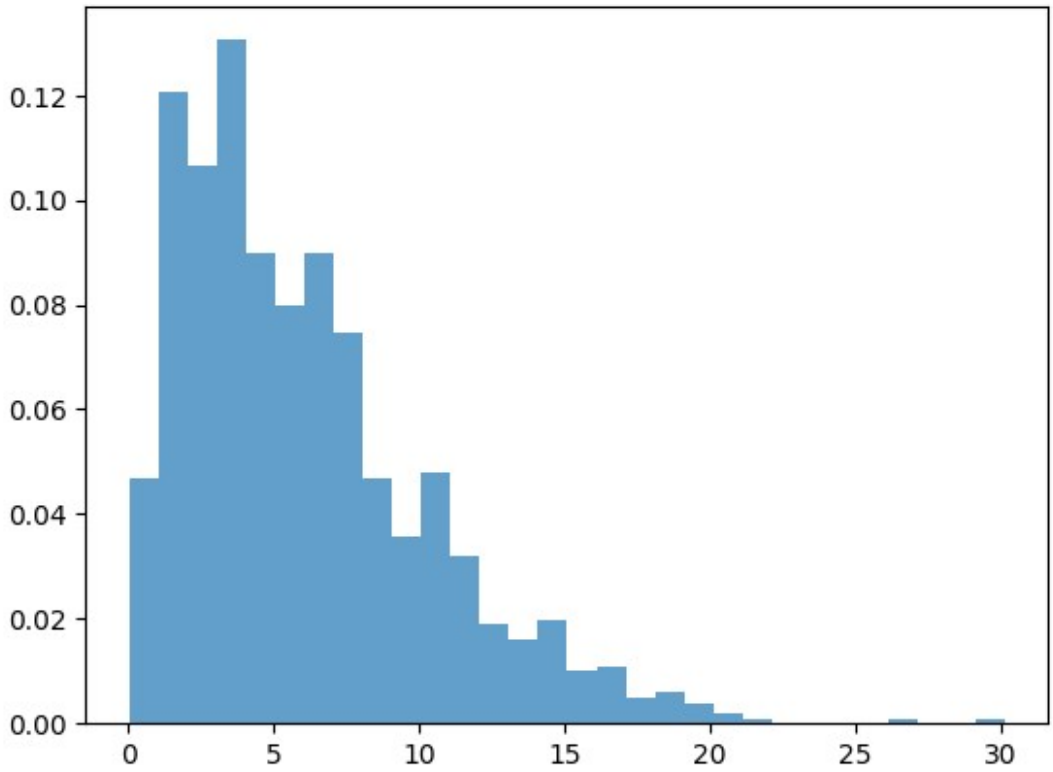
Ejercicio 6

Genere una muestra aleatoria de 1000 puntos de datos a partir de una distribución gamma con parámetro de forma 2 y parámetro de escala 3.

1. Utilice MLE para estimar los parámetros de forma y escala de la distribución gamma de la muestra.
2. ¿Qué tan cerca están los parámetros estimados de los parámetros verdaderos?
3. ¿Puede evaluar la bondad de ajuste de la distribución estimada a los datos observados?

```
shape, scale = 2, 3.
s = np.random.gamma(shape, scale, 1000)
params = stats.gamma.fit(s)

plt.hist(s, bins=30, density=True, alpha=0.7)

shape_est = params[0]
loc_est = params[1]
scale_est = params[2]

print("Forma estimada:" +str(shape_est))
print("Escala estimada:" +str(scale_est))

error_shape = abs((2 - shape_est) / 2) * 100
error_scale = abs((3 - scale_est) / 3) * 100

print("Error de forma:", shape_est, "%")
print("Error de escala:", scale_est, "%")

ks_statistic, p_value = stats.kstest(s, 'gamma', args=params)

print("Estadístico KS:", ks_statistic)
print("Valor p:", p_value)
```

```
Forma estimada:1.9410213862537766
Escala estimada:3.1482495331672307
Error de forma: 1.9410213862537766 %
Error de escala: 3.1482495331672307 %
Estadístico KS: 0.020741412578661134
Valor p: 0.7746544782925536
```



- Como se puede observar en los resultados anteriores, la forma y la escala estan bastante cercanos de los parámetros verdaderos. Se obtuvo una forma estimada de 1.9 siendo la forma verdera de 2, con un error de 2%. Asimismo se obtuvo una escala de 3.03 siendo la escala verdadera de 3, con error de 3%.
- El estadístico KS mide la máxima distancia vertical entre la función de distribución acumulada de la distribución estimada y los datos observados. En este caso, el valor del estadístico KS es relativamente bajo, lo que indica una buena concordancia entre la distribución estimada y los datos observados.

El valor p asociado es 0.4698578586221325, lo cual es mayor que un nivel de significancia comúnmente utilizado, como 0.05. Esto sugiere que no hay suficiente evidencia para rechazar la hipótesis nula de que los datos provienen de la distribución gamma estimada. En otras palabras, los datos observados se ajustan razonablemente bien a la distribución gamma estimada.

Por lo tanto, en base a estos resultados, podemos concluir que la distribución gamma estimada se ajusta de manera adecuada a los datos observados.

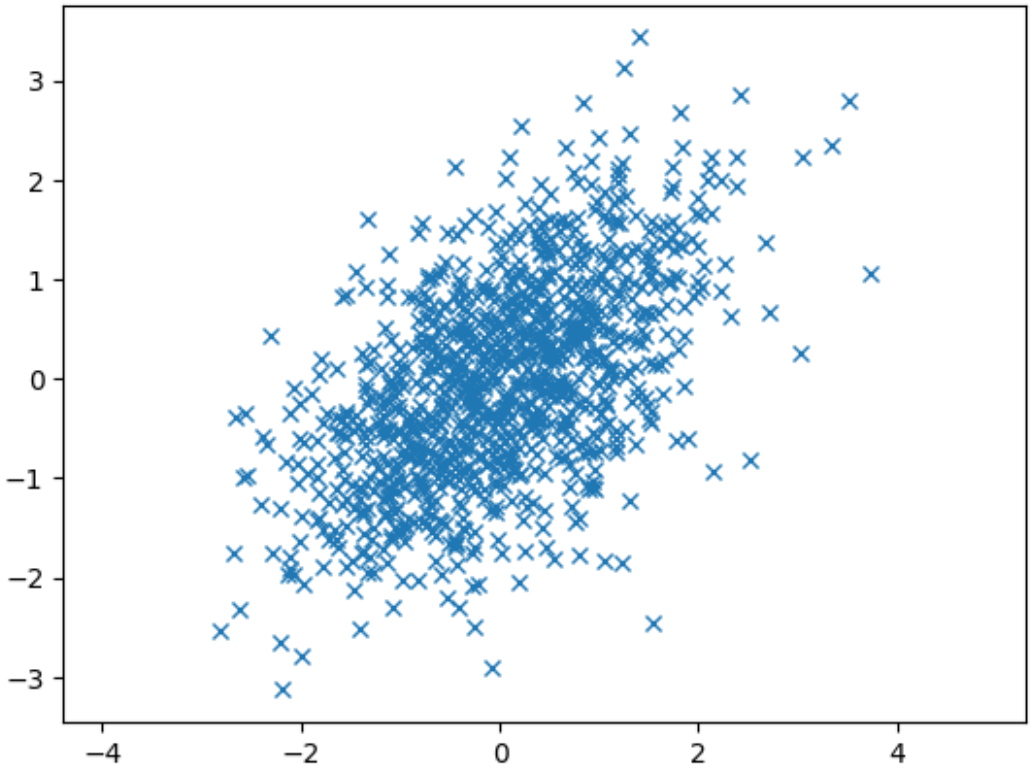
Ejercicio 7

Genere una muestra aleatoria de 1000 puntos de datos a partir de una distribución normal bivariada con vector medio [0, 0] y matriz de covarianza [[1, 0,5], [0,5, 1]]. Tasks:

1. Visualice los puntos de datos en un diagrama de dispersión.
2. Calcula e imprime el coeficiente de correlación entre las dos variables.

3. ¿Qué información puede obtener sobre la relación entre las variables del diagrama de dispersión y el coeficiente de correlación?

```
mean = [0, 0]
cov = [[1,0.5], [0.5,1]]
pts = np.random.multivariate_normal(mean, cov, 1000)
x, y = pts.T
plt.plot(x, y, 'x')
plt.axis('equal')
plt.show()

print("Coeficiento de correlación: " + str(np.corrcoef(pts.T)[0, 1]))
```




```
Coeficiento de correlación: 0.5497886280874528
```

El coeficiente de correlación indica la fuerza y la dirección de la relación lineal entre las variables. En este caso, el coeficiente de correlación es positivo y cercano a 0.5, lo que sugiere una correlación moderada positiva entre las dos variables. Asismo se puede ver una tendencia en los puntos a seguir una relacion lineal positiva.

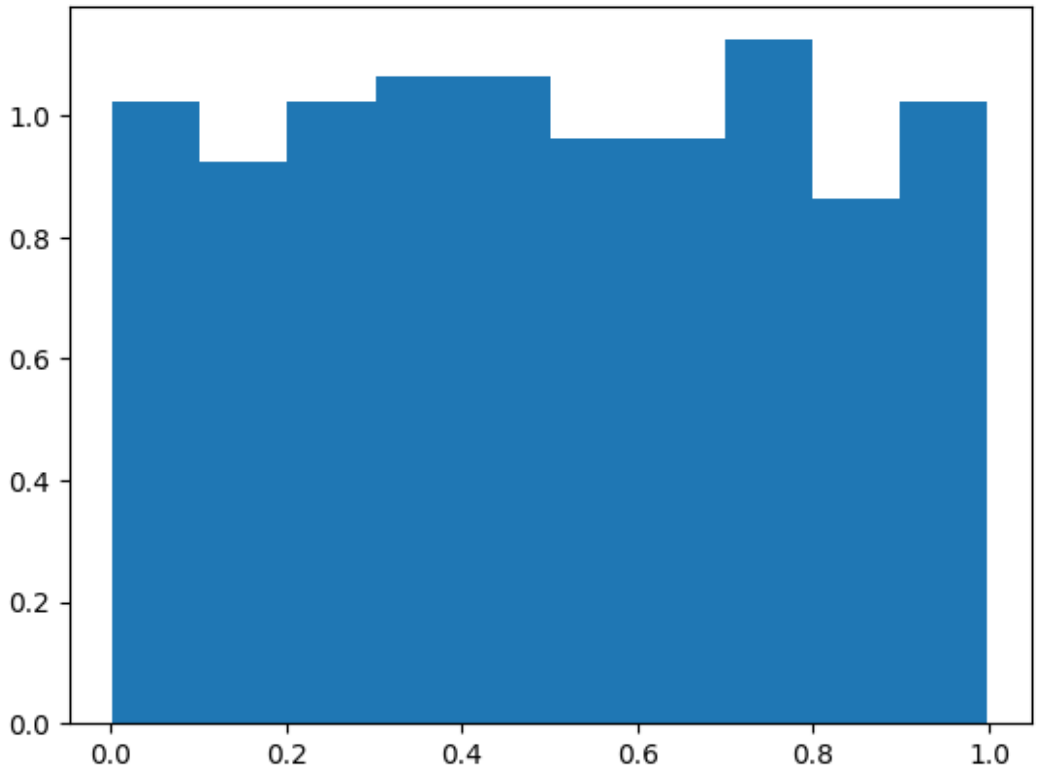
Ejercicio 8

Investigue en qué consiste la prueba “Goodness-of-Fit”. Luego, genere una muestra aleatoria de 500 puntos de datos a partir de una distribución uniforme entre 0 y 1. Tasks:

1. Realice una prueba de bondad de ajuste de chi-cuadrado para evaluar si la muestra sigue una distribución uniforme.
2. Interprete el resultado de la prueba y saque conclusiones sobre el ajuste de los datos.
3. ¿Puede sugerir alguna modificación para mejorar la bondad de ajuste?

```
from scipy.stats import chi2

def goodness_of_fit_uniform(data):
    expected_distribution = np.array([0.1, 0.1, 0.1, 0.1, 0.1, 0.1,
0.1, 0.1, 0.1, 0.1])
    observed_frequencies, _ = np.histogram(data, bins=10)

    total_samples = len(data)
    expected_frequencies = expected_distribution * total_samples

    chi2_stat = np.sum((observed_frequencies -
expected_frequencies)**2 / expected_frequencies)

    degrees_of_freedom = len(expected_distribution) - 1
    p_value = 1 - chi2.cdf(chi2_stat, degrees_of_freedom)

    return chi2_stat, p_value

sample_size = 500
data = np.random.uniform(0, 1, sample_size)
x = plt.hist(data, bins = 10, density = True)

chi2_stat, p_value = goodness_of_fit_uniform(data)

print("Chi-cuadraro:", chi2_stat)
print("valor de p:", p_value)
```

```
Chi-cuadraro: 2.6
valor de p: 0.9780720677116449
```



- La prueba de bondad de ajuste, también conocida como prueba "Goodness-of-Fit" en inglés, es una prueba estadística utilizada para determinar si una muestra de datos sigue una distribución teórica específica. En este caso, se desea evaluar si una muestra aleatoria de 500 puntos de datos sigue una distribución uniforme entre 0 y 1.

Referencia: https://medium.com/@moonchangin/how-to-do-goodness-of-fit-test-in-python-fe33b4d37ea2#:~:text=The%20chi%2Dsquare%20(%CF%87%C2%B2),function%20from%20the%20scipy%20module.

- El valor de p obtenido en la prueba de chi-cuadrado indica la probabilidad de obtener un estadístico de prueba igual o más extremo que el observado, asumiendo que los datos siguen una distribución uniforme entre 0 y 1. En este caso, el valor de p es bastante alto (0.978). Por lo tanto, se puede concluir que los datos muestrales siguen una distribución uniforme entre 0 y 1.
- Obtener una muestra más grande podría proporcionar una evaluación más precisa del ajuste. Asimismo se puede aumentar la cantidad de frecuencias.
