TP 2.2 - Generadores de números pseudoaleatorios de distintas Distribuciones de Probabilidad.

Gabriel Dequelli
Universidad Tecnológica Nacional - FRRO
Zeballos 1341, S2000, Argentina
gdequelli98@gmail.com

Ignacio Iturburu
Universidad Tecnológica Nacional - FRRO
Zeballos 1341, S2000, Argentina
ignacioiturburu07@gmail.com

Juan Manuel Tarrago
Universidad Tecnológica Nacional - FRRO
Zeballos 1341, S2000, Argentina
juanmanueltarrago@gmail.com

June 10, 2024

Abstract

Los generadores de números pseudoaleatorios (PRNG) son herramientas fundamentales en diversos campos como la criptografía, las simulaciones y la modelización estadística. Este artículo presenta un estudio de los PRNG diseñados para producir números que siguen diferentes distribuciones de probabilidad, incluyendo las distribuciones uniforme, normal, exponencial y distribuciones personalizadas. Exploramos las bases teóricas de los PRNG y sus estrategias de implementación, enfatizando los métodos utilizados para transformar números aleatorios uniformemente distribuidos en otras distribuciones. Se discuten en detalle algoritmos clave como el método de muestreo por transformada inversa y la técnica de muestreo por rechazo. La eficiencia de estos generadores se evalúa en función de pruebas estadísticas. Nuestros resultados destacan las compensaciones entre precisión, velocidad y complejidad en la generación de números pseudoaleatorios en varias distribuciones. Este estudio tiene como objetivo proporcionar un marco sólido para la selección e implementación de PRNG adaptados a aplicaciones específicas, mejorando la fiabilidad y la eficiencia de las simulaciones y análisis probabilísticos.

1 Introducción

El estudio anterior demostró que la generación de números pseudoaleatorios no es una tarea sencilla, pero es esencial para nuestras necesidades. Afortunadamente, los generadores de números pseudoaleatorios (PRNGs) han sido desarrollados y probados, proporcionándonos valores uniformes continuos entre 0 y 1. Sin embargo, en aplicaciones de Probabilidad y Estadística, a menudo necesitamos valores distribuidos de acuerdo con diferentes distribuciones de probabilidad, tanto continuas como discretas.

La pregunta es: £cómo generamos valores según distintas distribuciones? La solución a este problema ha sido desarrollada previamente y está bien documentada. Nuestra tarea es redescubrir e implementar estos mecanismos para nuestros experimentos.

El objetivo de este trabajo es construir generadores de números pseudoaleatorios para varias distribuciones de probabilidad, siguiendo las directrices y métodos del autor Thomas Naylor. Este estudio incluye la implementación teórica y práctica de generadores para diferentes distribuciones.

2 Descripción del trabajo

El trabajo consiste en un estudio detallado de varias distribuciones de probabilidad que son esenciales para la simulación de fenómenos de interés. Para cada distribución, se deben realizar las siguientes tareas:

Introducción teórica de la distribución de probabilidad: Descripción de la distribución y sus parámetros. Función de probabilidad (para distribuciones discretas) o función de densidad (para distribuciones continuas). Gráficas ilustrativas.

Desarrollo matemático: Derivación de la función de distribución acumulada (CDF). Obtención de la inversa de la CDF (si es posible), que se utilizará para generar valores distribuidos de acuerdo con la distribución deseada.

Implementación en Python: Elaboración de un programa en Python 3.x para generar números pseudoaleatorios según la distribución estudiada.

Testeo y validación: Implementación de pruebas adecuadas para verificar la correcta generación de valores según la distribución. La metodología de testeo queda a criterio del grupo, pudiendo incluir pruebas gráficas, pruebas estadísticas, entre otras.

Generadores para distintas distribuciones de probabilidad:

3 Distribución Uniforme

3.1 Introducción Teórica

La función de densidad de probabilidad más simple es aquella que se caracteriza por ser constante en el intervalo (a, b) y cero fuera de él. Esta función de densidad define la distribución uniforme. La distribución uniforme surge cuando se estudian características de los errores por redondeo al registrar un conjunto de medidas sujetas a cierto nivel de precisión.

3.2 Parámetros

- a: límite inferior.
- b: límite superior.

3.3 Función de densidad de probabilidad (PDF)

Matemáticamente, la función de densidad uniforme se define como sigue:

$$f(x) = \begin{cases} \frac{1}{b-a} & \text{si } a \leq x \leq b, \\ 0 & \text{fuera del intervalo } (a, b). \end{cases}$$

3.4 Gráfica de la función

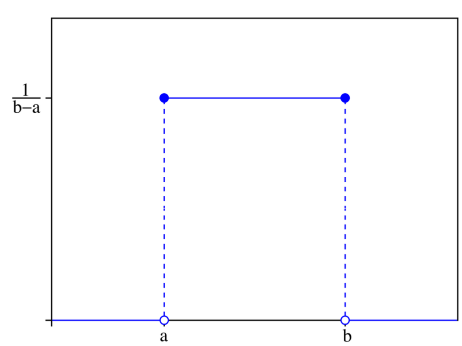

Figure 1: Gráfica de la distribución uniforme

3.5 Función de distribución acumulada (CDF)

La función de la distribución acumulativa $F(x)$, para una variable aleatoria X uniformemente distribuida, se puede representar de la siguiente forma:

$$F(x) = \begin{cases} 0 & \text{si } x < a, \\ \frac{x-a}{b-a} & \text{si } a \leq x \leq b, \\ 1 & \text{si } x > b. \end{cases}$$

3.6 Desarrollo Matemático

Para generar valores distribuidos uniformemente en el intervalo $[a, b]$, podemos utilizar dos métodos: Por medio de la Transformada Inversa o a través del Método del Rechazo

3.6.1 Transformada Inversa

Para realizar el cálculo, buscamos la inversa de la CDF $F(x)$ la cual resultó ser:

$$F^{-1}(u) = a + u(b - a), \quad \text{donde } u \in [0, 1].$$

3.6.2 Método del rechazo

Para este caso utilizaremos debemos utilizar otra función $g(x)$ que pertenezca a una distribución sencilla, y una constante c tal que $f(x) \leq c * g(x)$ para todo x. En este caso elegimos $g(x) = \frac{1}{b-a}$ y $c = 1$.

3.7 Testeo y Validación

Para testear la generación de valores, utilizamos histogramas para visualizar la distribución de los valores generados y compararla con la generación hecha por la biblioteca NumPy (utilizada en los algoritmos anteriores) de Python.

Las gráficas obtenidas generadas por el código fueron las siguientes:

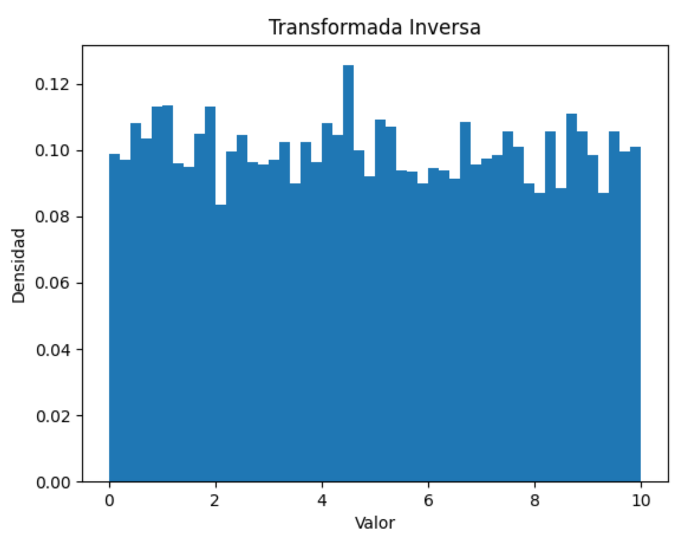

Figure 2: Histograma de valores uniformemente distribuidos generados por la inversa

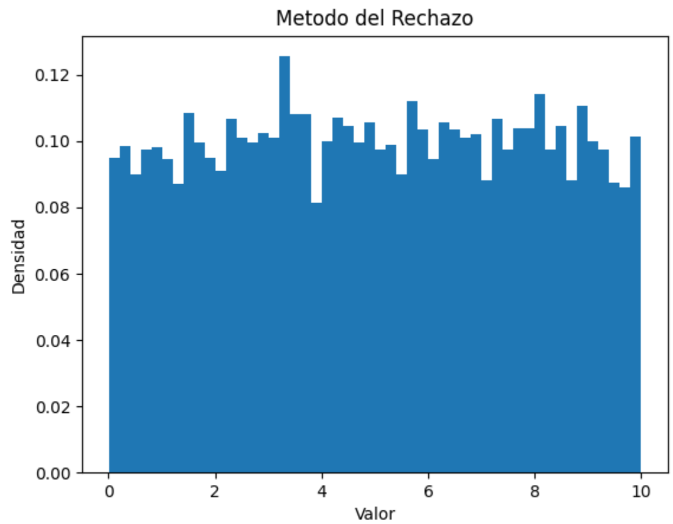

Figure 3: Histograma de valores uniformemente distribuidos generados por el método del rechazo

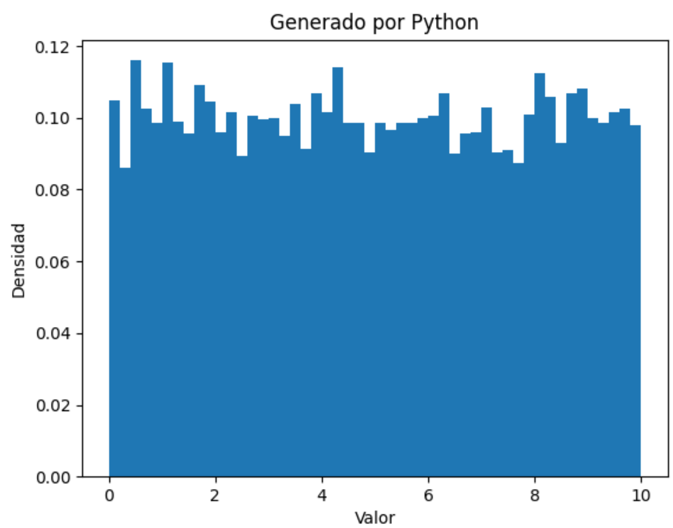

Figure 4: Histograma de valores uniformemente distribuidos generados por la biblioteca NumPy de Python

3.8 Implementación en Python

```python
import numpy as np

def generar_uniforme(a, b, size=1):
    u = np.random.uniform(0, 1, size)
    return a + u * (b - a)
```

Código 1: Generador de números uniformemente distribuidos

4 Distribución Exponencial

4.1 Introducción Teórica

La distribución exponencial es una distribución de probabilidad continua que describe el tiempo entre eventos en un proceso de Poisson..

4.2 Parámetros

- λ: tasa de ocurrencia de eventos.

4.3 Función de densidad de probabilidad (PDF)

La función de densidad de probabilidad de la distribución exponencial se define como:

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & \text{si } x \geq 0, \\ 0 & \text{si } x < 0. \end{cases}$$

4.4 Gráfica de la función

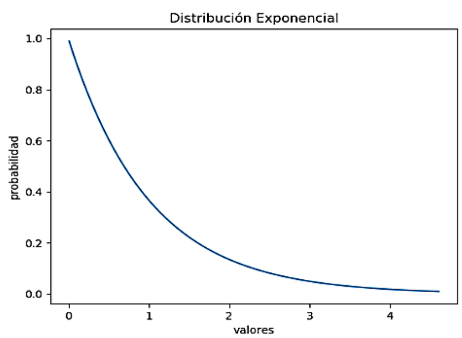

Figure 5: Gráfica de la distribución exponencial

4.5 Función de distribución acumulada (CDF)

La función de distribución acumulada de la distribución exponencial se define como:

$$F(x) = \begin{cases} 1 - e^{-\lambda x} & \text{si } x \geq 0, \\ 0 & \text{si } x < 0. \end{cases}$$

4.6 Desarrollo Matemático

Para generar valores distribuidos exponencialmente, podemos utilizar dos métodos: Por medio de la Transformada Inversa o a través del Método del Rechazo.

4.6.1 Transformada Inversa

Para realizar el cálculo, buscamos la inversa de la CDF $F(x)$ la cual resultó ser:

$$F^{-1}(u) = -\frac{1}{\lambda} \ln(1-u), \quad \text{donde } u \in [0, 1].$$

4.6.2 Método del rechazo

Para este caso utilizaremos debemos utilizar otra función $g(x)$ que pertenezca a una distribución sencilla, y una constante c tal que $f(x) \leq c * g(x)$ para todo x. En este caso elegimos $g(x) = \lambda e^{-\lambda x}$ y $c = 1$.

4.7 Testeo y Validación

Para testear la generación de valores, utilizamos histogramas para visualizar la distribución de los valores generados y compararla con la generación hecha por la biblioteca NumPy (utilizada en los algoritmos anteriores) de Python. Las gráficas obtenidas generadas por el código fueron las siguientes:

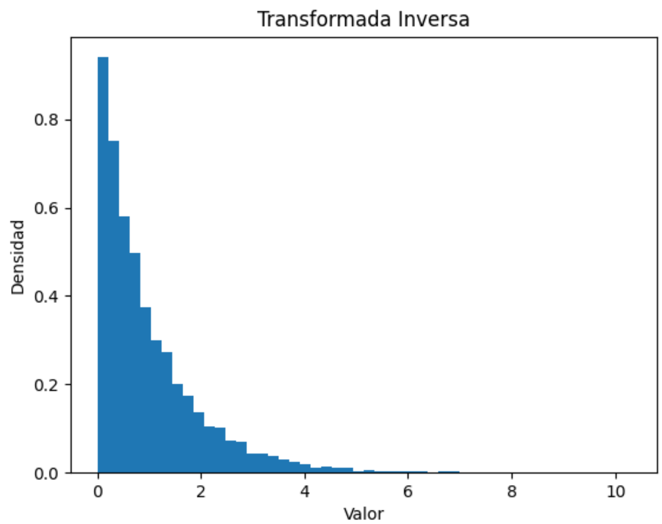

Figure 6: Histograma de valores exponencialmente distribuidos generados por la inversa

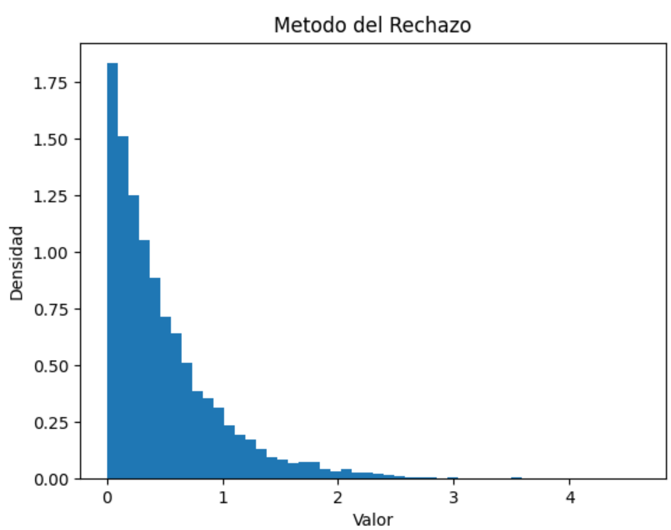

Figure 7: Histograma de valores exponencialmente distribuidos generados por el método del rechazo

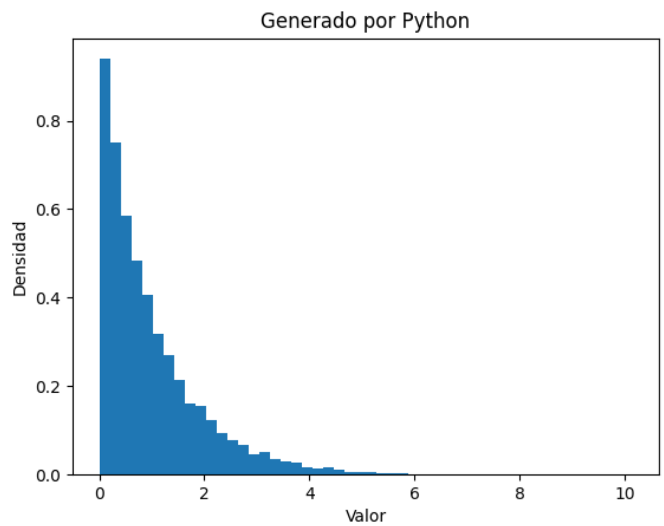

Figure 8: Histograma de valores exponencialmente distribuidos generados por la biblioteca NumPy de Python

5 Distribución Normal

5.1 Introducción Teórica

La distribución normal es una de las distribuciones de probabilidad más importantes en estadística. Se caracteriza por ser simétrica y centrada en su media, con una forma de campana.

5.2 Parámetros

- μ: media.
- σ: desviación estándar.

5.3 Función de densidad de probabilidad (PDF)

La función de densidad de probabilidad de la distribución normal se define como:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}.$$

5.4 Gráfica de la función

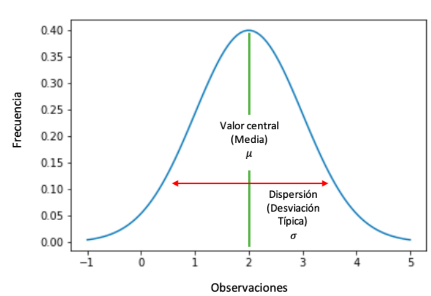

Figure 9: Gráfica de la distribución normal

5.5 Función de distribución acumulada (CDF)

La función de distribución acumulada de la distribución normal se define como:

$$F(x) = \frac{1}{2}\left[1 + \operatorname{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right],$$

donde $\operatorname{erf}(x)$ es la función de error.

5.6 Desarrollo Matemático

Para generar valores distribuidos normalmente, podemos utilizar dos métodos: Por medio de la Transformada Inversa o a través del Método del Rechazo.

5.6.1 Transformada Inversa

Para realizar el cálculo, buscamos la inversa de la CDF $F(x)$ la cual resultó ser:

$$F^{-1}(u) = \mu + \sigma\sqrt{2}\operatorname{erf}^{-1}(2u-1), \quad \text{donde } u \in [0,1].$$

5.6.2 Método del rechazo

Para este caso utilizaremos debemos utilizar otra función $g(x)$ que pertenezca a una distribución sencilla, y una constante c tal que $f(x) \leq c * g(x)$ para todo x. En este caso elegimos $g(x) = \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$ y $c = 1$.

5.7 Testeo y Validación

Para testear la generación de valores, utilizamos histogramas para visualizar la distribución de los valores generados y compararla con la generación hecha por la biblioteca NumPy (utilizada en los algoritmos anteriores) de Python.

Las gráficas obtenidas generadas por el código fueron las siguientes:

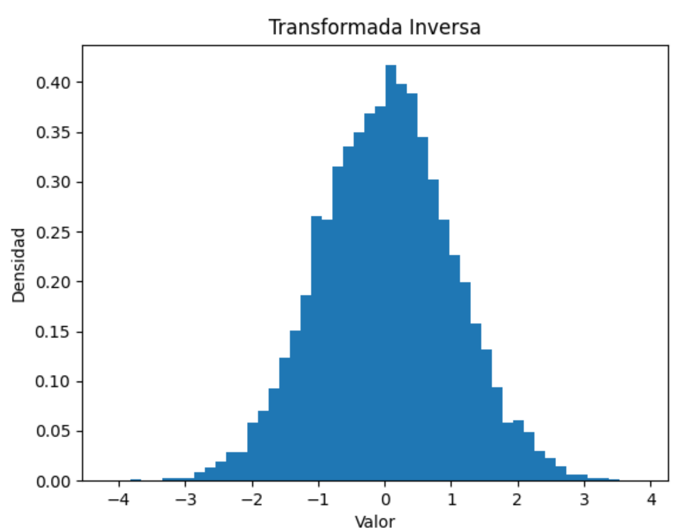

Figure 10: Histograma de valores normalmente distribuidos generados por la inversa

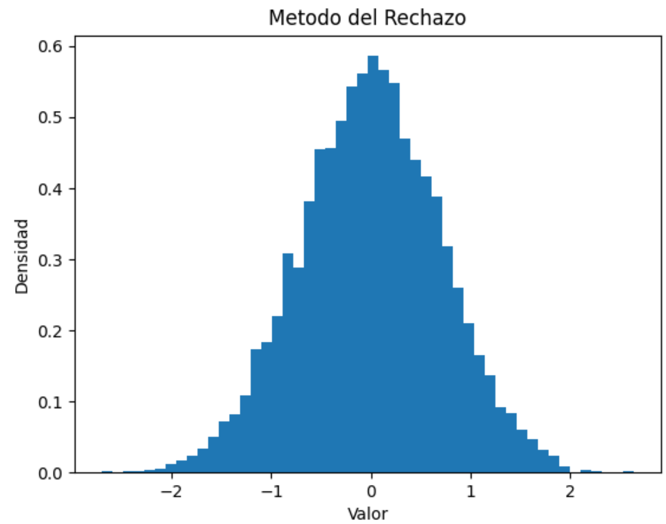

Figure 11: Histograma de valores normalmente distribuidos generados por el método del rechazo

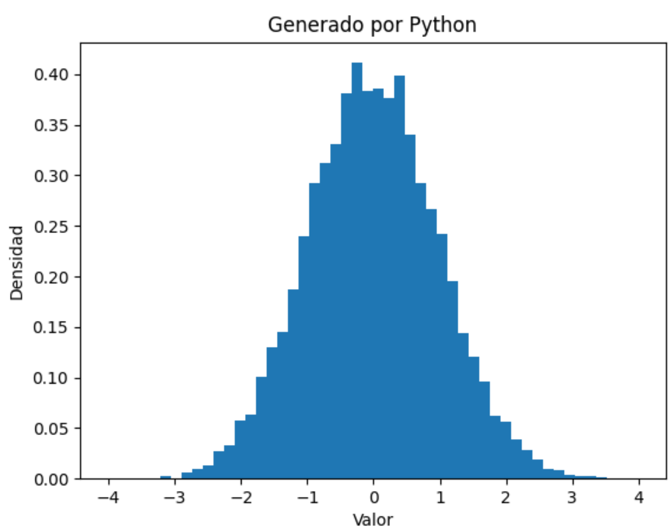

Figure 12: Histograma de valores normalmente distribuidos generados por la biblioteca NumPy de Python

```python
import numpy as np

def generar_normal(mu, sigma, size=1):
    u1 = np.random.uniform(0, 1, size)
    u2 = np.random.uniform(0, 1, size)
    z0 = np.sqrt(-2 * np.log(u1)) * np.cos(2 * np.pi * u2)
    return mu + z0 * sigma
```

Código 2: Generador de números normalmente distribuidos

6 Distribución Gamma

6.1 Introducción Teórica

La distribución de gamma es una distribución de probabilidad continua que se utiliza para modelar el tiempo de espera hasta que ocurran un número específico de eventos. Es una distribución de dos parámetros, lo que significa que requiere dos parámetros para definir completamente la distribución. Los dos parámetros son el parámetro de forma, denotado por α, y el parámetro de escala, denotado por β. El parámetro de forma controla la forma de la distribución, mientras que el parámetro de escala controla la propagación de la distribución.

6.2 Parámetros

- α: parámetro de forma.
- β: parámetro de escala.

6.3 Función de densidad de probabilidad (PDF)

La función de densidad de probabilidad de la distribución gamma se define como:

$$f(x) = \frac{\beta^\alpha x^{\alpha-1} e^{-\beta x}}{\Gamma(\alpha)},$$

donde $\Gamma(\alpha)$ es la función gamma.

6.4 Gráfica de la función

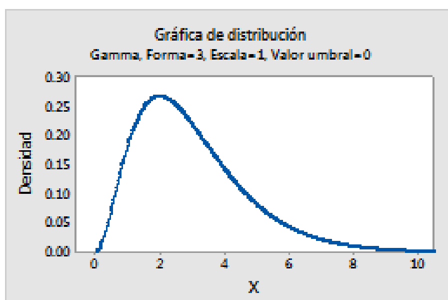

Figure 13: Gráfica de la distribución gamma

6.5 Función de distribución acumulada (CDF)

La función de distribución acumulada de la distribución gamma se define como:

$$F(x) = \frac{1}{\Gamma(\alpha)} \gamma(\alpha, \beta x),$$

donde $\gamma(\alpha, x)$ es la función gamma incompleta.

6.6 Desarrollo Matemático

Para generar valores distribuidos gamma, podemos utilizar dos métodos: Por medio de la Transformada Inversa o a través del Método del Rechazo.

6.7 Método del rechazo

Para este caso utilizaremos debemos utilizar otra función $g(x)$ que pertenezca a una distribución sencilla, y una constante c tal que $f(x) \leq c * g(x)$ para todo x. En este caso elegimos $g(x) = \frac{\beta^\alpha x^{\alpha-1} e^{-\beta x}}{\Gamma(\alpha)}$ y $c = 1$.

7 Distribución de Poisson

7.1 Introducción Teórica

La distribución de Poisson es una distribución de probabilidad discreta que describe el número de eventos que ocurren en un intervalo de tiempo fijo o en un área fija. Concretamente, se especializa en la probabilidad de ocurrencia de sucesos con probabilidades muy pequeñas, o sucesos raros.

7.2 Parámetros

- λ: tasa de ocurrencia de eventos.

7.3 Función de densidad de probabilidad (PDF)

La función de densidad de probabilidad de la distribución de Poisson se define como:

$$f(x) = \frac{e^{-\lambda}\lambda^x}{x!}.$$

7.4 Gráfica de la función

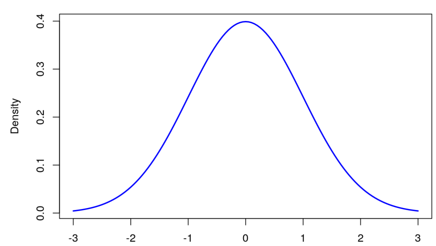

Figure 14: Gráfica de la distribución de Poisson

7.5 Función de distribución acumulada (CDF)

La función de distribución acumulada de la distribución de Poisson se define como:

$$F(x) = e^{-\lambda}\sum_{i=0}^{x}\frac{\lambda^i}{i!}.$$

7.6 Desarrollo Matemático

Para generar valores distribuidos de Poisson, podemos utilizar dos métodos: Por medio de la Transformada Inversa o a través del Método del Rechazo.

7.6.1 Método del rechazo

Para este caso utilizaremos debemos utilizar otra función $g(x)$ que pertenezca a una distribución sencilla, y una constante c tal que $f(x) \leq c * g(x)$ para todo x. En este caso elegimos $g(x) = \frac{e^{-\lambda}\lambda^x}{x!}$ y $c = 1$.

7.7 Testeo y Validación

Para testear la generación de valores, utilizamos histogramas para visualizar la distribución de los valores generados y compararla con la generación hecha por la biblioteca NumPy (utilizada en los algoritmos anteriores) de Python.

Las gráficas obtenidas generadas por el código fueron las siguientes:

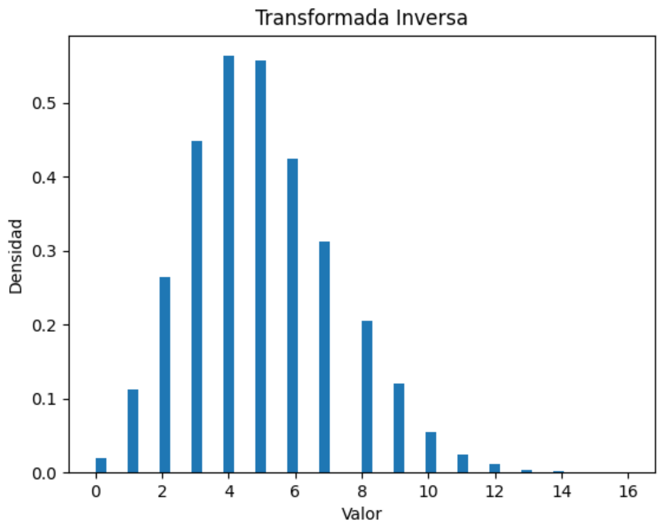

Figure 15: Histograma de valores de Poisson generados por la inversa

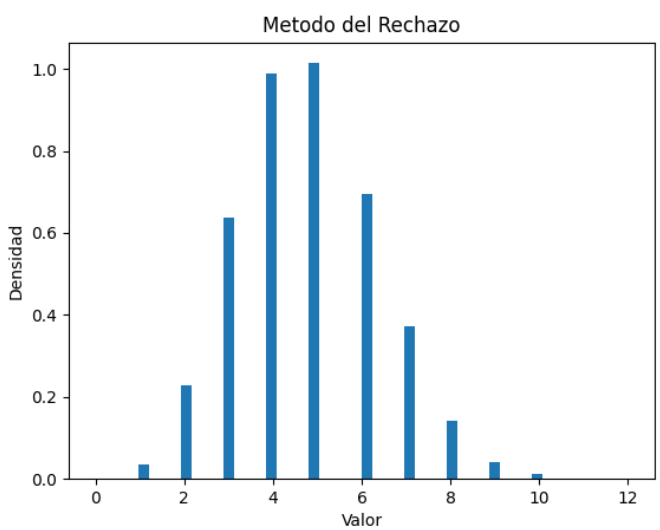

Figure 16: Histograma de valores de Poisson generados por el método del rechazo

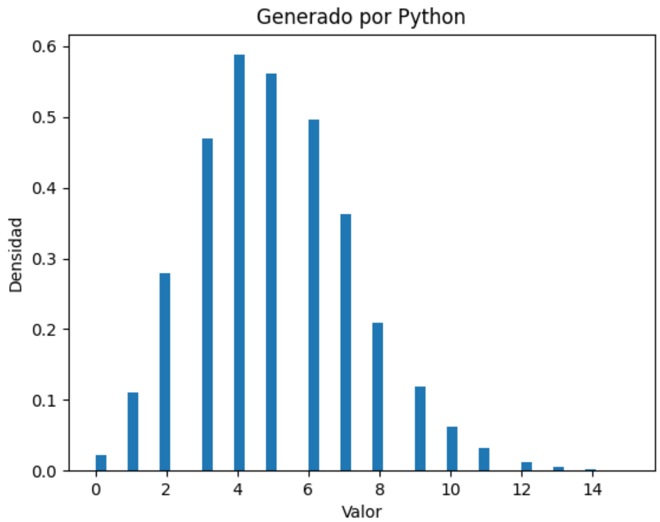

Figure 17: Histograma de valores de Poisson generados por la biblioteca NumPy de Python

8 Distribución de Pascal

8.1 Introducción Teórica

La distribución de Pascal es una distribución de probabilidad discreta que describe el número de ensayos necesarios para obtener un número fijo de éxitos en un proceso de Bernoulli. Esta distribución es útil en situaciones donde se modelan procesos de conteo y el objetivo es obtener un número específico de éxitos.

8.2 Parámetros

- r: número de éxitos.
- p: probabilidad de éxito.

8.3 Función de densidad de probabilidad (PDF)

La función de densidad de probabilidad de la distribución de Pascal se define como:

$$f(x) = \binom{x-1}{r-1} p^r (1-p)^{x-r}.$$

8.4 Gráfica de la función

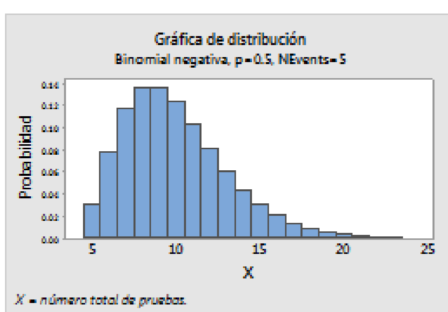

Figure 18: Gráfica de la distribución de Pascal

8.5 Función de distribución acumulada (CDF)

La función de distribución acumulada de la distribución de Pascal se define como:

$$F(x) = I_{1-p}(r, x - r + 1),$$

donde $I_{1-p}(r, x - r + 1)$ es la función beta incompleta regularizada.

8.6 Desarrollo Matemático

Para generar valores distribuidos de Pascal, podemos utilizar dos métodos: Por medio de la Transformada Inversa o a través del Método del Rechazo.

8.7 Método del rechazo

Para este caso utilizaremos debemos utilizar otra función $g(x)$ que pertenezca a una distribución sencilla, y una constante c tal que $f(x) \leq c * g(x)$ para todo x. En este caso elegimos $g(x) = \binom{x-1}{r-1} p^r (1-p)^{x-r}$ y $c = 1$.

9 Distribución Binomial

9.1 Introducción Teórica

La distribución binomial es una distribución de probabilidad discreta que cuenta el número de éxitos en una secuencia de n ensayos de Bernoulli independientes entre sí con una probabilidad fija p de ocurrencia de éxito entre los ensayos.

9.2 Parámetros

- n: número de ensayos.
- p: probabilidad de éxito.

9.3 Función de densidad de probabilidad (PDF)

La función de densidad de probabilidad de la distribución binomial se define como:

$$f(x) = \binom{n}{x} p^x (1-p)^{n-x}.$$

9.4 Gráfica de la función

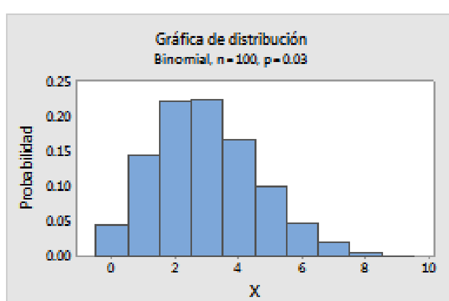

Figure 19: Gráfica de la distribución binomial

9.5 Función de distribución acumulada (CDF)

La función de distribución acumulada de la distribución binomial se define como:

$$F(x) = I_{1-p}(n-x, x+1),$$

donde $I_{1-p}(n-x, x+1)$ es la función beta incompleta regularizada.

9.6 Desarrollo Matemático

Para generar valores distribuidos de binomial, podemos utilizar dos métodos: Por medio de la Transformada Inversa o a través del Método del Rechazo.

9.6.1 Método del rechazo

Para este caso utilizaremos debemos utilizar otra función $g(x)$ que pertenezca a una distribución sencilla, y una constante c tal que $f(x) \leq c * g(x)$ para todo x. En este caso elegimos $g(x) = \binom{n}{x} p^x (1-p)^{n-x}$ y $c = 1$.

9.7 Testeo y Validación

Para testear la generación de valores, utilizamos histogramas para visualizar la distribución de los valores generados y compararla con la generación hecha por la biblioteca NumPy (utilizada en los algoritmos anteriores) de Python.

Las gráficas obtenidas generadas por el código fueron las siguientes:

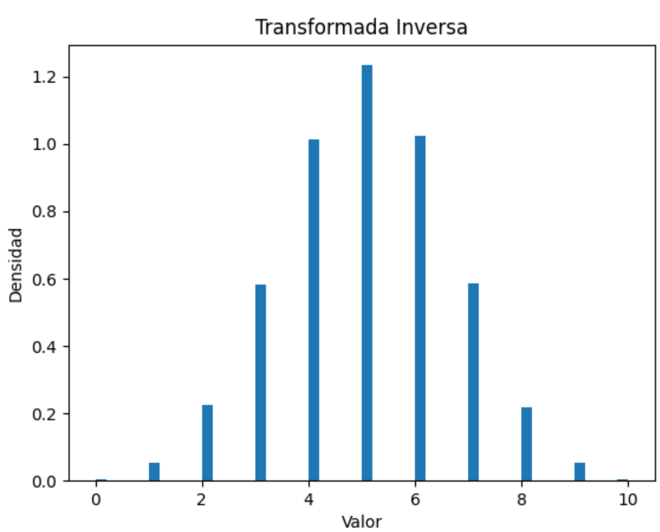

Figure 20: Histograma de valores binomial generados por la inversa

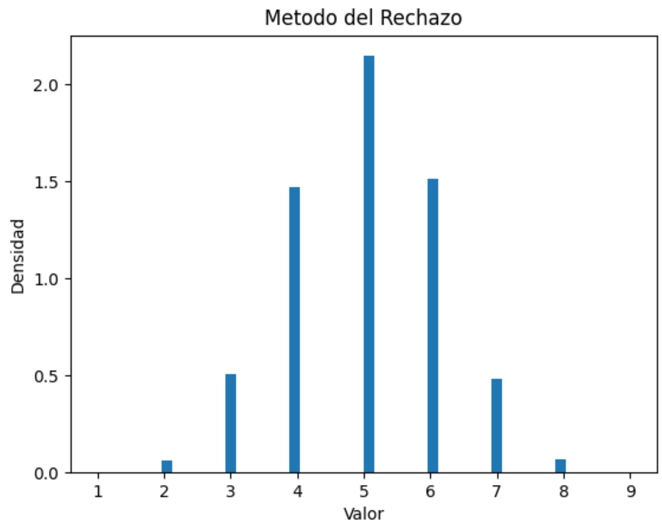

Figure 21: Histograma de valores binomial generados por el método del rechazo

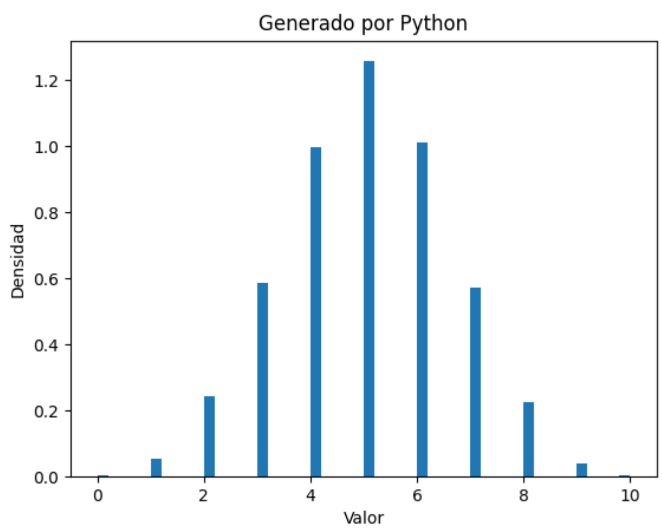

Figure 22: Histograma de valores binomial generados por la biblioteca NumPy de Python

10 Distribución Hipergeométrica

10.1 Introducción Teórica

La distribución hipergeométrica es una distribución de probabilidad discreta que describe el número de éxitos en una muestra de tamaño fijo sin reemplazo.

10.2 Parámetros

- N: tamaño de la población.
- n: número de éxitos en la población.
- m: tamaño de la muestra.

10.3 Función de densidad de probabilidad (PDF)

La función de densidad de probabilidad de la distribución hipergeométrica se define como:

$$f(x) = \frac{\binom{n}{x}\binom{N-n}{m-x}}{\binom{N}{m}}.$$

10.4 Gráfica de la función

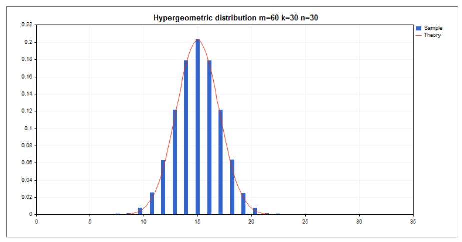

Figure 23: Gráfica de la distribución hipergeométrica

10.5 Función de distribución acumulada (CDF)

La función de distribución acumulada de la distribución hipergeométrica se define como:

$$F(x) = \sum_{i=0}^{x} \frac{\binom{n}{i}\binom{N-n}{m-i}}{\binom{N}{m}}.$$

10.6 Desarrollo Matemático

Para generar valores distribuidos de hipergeométrica, podemos utilizar dos métodos: Por medio de la Transformada Inversa o a través del Método del Rechazo.

10.6.1 Método del rechazo

Para este caso utilizaremos debemos utilizar otra función $g(x)$ que pertenezca a una distribución sencilla, y una constante c tal que $f(x) \leq c * g(x)$ para todo x. En este caso elegimos $g(x) = \frac{\binom{n}{x}\binom{N-n}{m-x}}{\binom{N}{m}}$ y $c = 1$.

11 Distribución empírica discreta

11.1 Introducción Teórica

La distribución empírica discreta es una distribución de probabilidad discreta que describe la frecuencia de ocurrencia de los valores de una muestra.

11.2 Parámetros

- x_i: valores de la muestra.
- p_i: frecuencia de ocurrencia de x_i.

11.3 Función de densidad de probabilidad (PDF)

La función de densidad de probabilidad de la distribución empírica discreta se define como:

$$f(x) = p_i, \quad \text{si } x = x_i.$$

11.4 Gráfica de la función

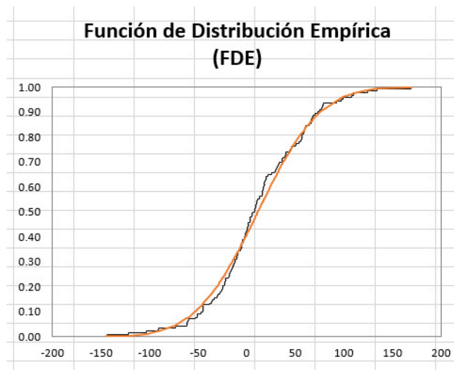

Figure 24: Gráfica de la distribución empírica discreta

11.5 Función de distribución acumulada (CDF)

La función de distribución acumulada de la distribución empírica discreta se define como:

$$F(x) = \sum_{x_i \leq x} p_i.$$

11.6 Desarrollo Matemático

Para generar valores distribuidos de empírica discreta, podemos utilizar dos métodos: Por medio de la Transformada Inversa o a través del Método del Rechazo.

11.6.1 Método del rechazo

Para este caso utilizaremos debemos utilizar otra función $g(x)$ que pertenezca a una distribución sencilla, y una constante c tal que $f(x) \leq c * g(x)$ para todo x. En este caso elegimos $g(x) = p_i$ y $c = 1$.

11.7 Testeo y Validación

Para testear la generación de valores, utilizamos histogramas para visualizar la distribución de los valores generados y compararla con la generación hecha por la biblioteca NumPy (utilizada en los algoritmos anteriores) de Python.

Las gráficas obtenidas generadas por el código fueron las siguientes:

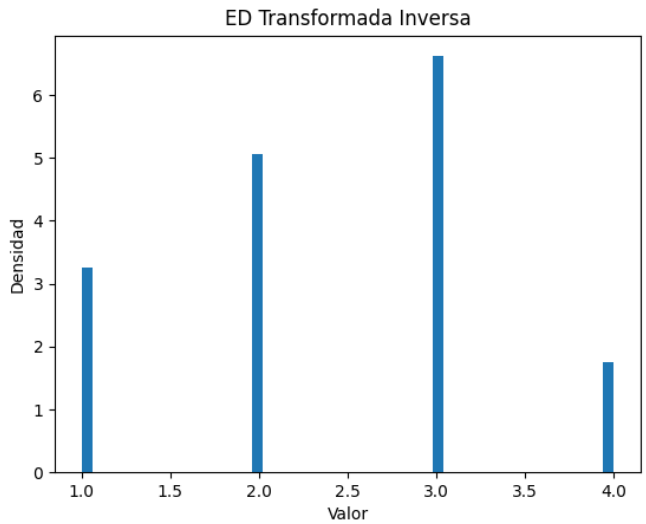

Figure 25: Histograma de valores empírica discreta generados por la inversa

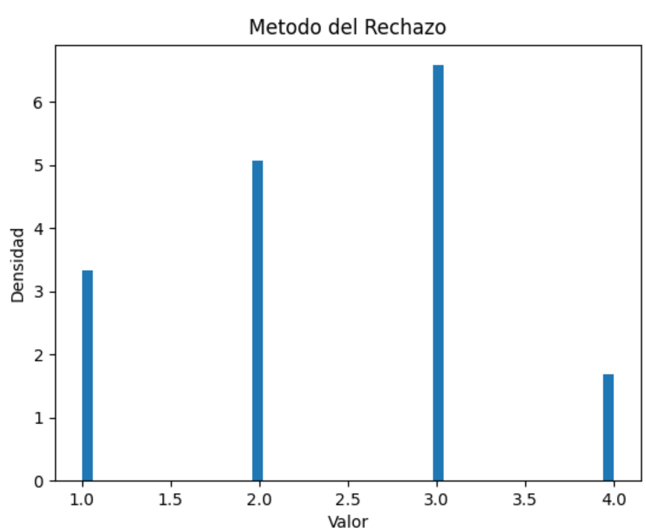

Figure 26: Histograma de valores empírica discreta generados por el método del rechazo

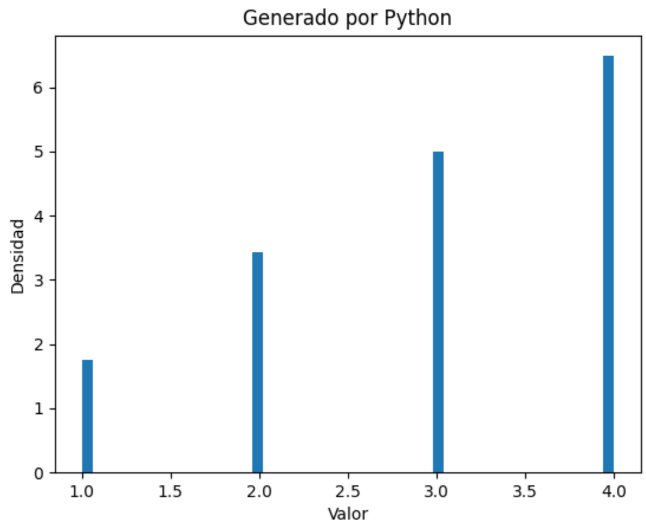

Figure 27: Histograma de valores empírica discreta generados por la biblioteca NumPy de Python

References

[1] Naylor, T.H. Técnicas de simulación en computadoras, 1982.

[2] Plantilla para la inserción de código Python
https://es.overleaf.com/project/666466f8febc5445875d7493

[3] Distribución binomial
https://es.wikipedia.org/wiki/Distribuci%C3%B3n_binomial