TP 2.2 - GENERADORES DE NÚMEROS PSEUDOALEATORIOS DE DISTINTAS DISTRIBUCIONES DE PROBABILIDAD

Gardeñes Fernando
Cátedra de Simulación - ISI
UTN - FRRo
Zeballos 1341, S2000 Rosario, Santa Fe
fergardenes7@gmail.com

Biscaldi Ivan
Cátedra de Simulación - ISI
UTN - FRRo
Zeballos 1341, S2000 Rosario, Santa Fe
ibiscaldi@frro.utn.edu.ar

Saludas Fausto
Cátedra de Simulación - ISI
UTN - FRRo
Zeballos 1341, S2000 Rosario, Santa Fe
fausaludas14@gmail.com

Fierro Nicolás
Cátedra de Simulación - ISI
UTN - FRRo
Zeballos 1341, S2000 Rosario, Santa Fe
nicofierro1@gmail.com

30 de mayo de 2023

ABSTRACT

En el presente informe se exploran y evalúan generadores basados en distribuciones Uniforme, Gamma, Exponencial y Normal; y Binomial, Geométrica, Hipergeométrica y Poisson. Se discuten las propiedades de cada generador, como la calidad de los números generados, la eficiencia computacional y la facilidad de implementación. Se implementan algoritmos y se realizan experimentos para generar secuencias de números aleatorios con cada distribución. Este trabajo contribuye a la comprensión y aplicación de generadores de números aleatorios en el ámbito de la simulación y la estadística.

Palabras claves: Variable aleatoria, Distribución de probabilidad, función de densidad, Esperanza Matemática, Varianza, Python 3, Distribución Uniforme, Distribución Normal, Distribución Higrométrica, Distribución de Poisson, Distribución Binomial, Distribución de Pascal, Distribución Gamma, Distribución de Erlang, Distribución Exponencial, Distribución Empírica

1. Introducción

El modelizado de sistemas complejos para llevar acabo experimentos es uno de los pasos claves de la simulación. Para poder modelar el funcionamiento de un sistema pueden requerirse números que, aunque deben ser aleatorios, para poder representar un comportamiento deben seguir una determinada distribución. Por ello abordaremos la generación de números aleatorios distribuidos.

Primero se proporcionara un marco teórico para comprender el comportamiento de las distintas distribuciones, y posteriormente abordaremos la generación de números distribuidos.

2. Marco Teórico

Variable Aleatoria: Una variable aleatoria (v.a)[2] **X** es una función que asocia a cada suceso elemental del espacio muestral **E** de un experimento aleatorio un valor numérico real.

$$X : E \to R$$

X se puede clasificar como:

- **v.a discreta**: X es una variable aleatoria discreta si se puede contar el conjunto de valores posibles, o sus valores posibles son una serie interminable de números con tantos elementos como números enteros existen.
- **v.a continua**: X es una variable aleatoria continua si puede tomar valores en una escala continua, es decir, X puede tomar cualquiera de los infinitos valores que se encuentran en cualquier intervalo de números reales.

Algunas variables aleatorias con distribución de probabilidad de tipo continua:

- Uniforme
- Exponencial
- Normal
- Gamma

Algunas variables aleatorias con distribución de probabilidad de tipo discreta:

- Binomial
- Hipergeométrica
- Pascal (Binomial negativa)
- Poisson

Una variable aleatoria de cierta distribución tiene asociada una función de distribución acumulada (FDA) y si la variable es continua también tendrá una función de densidad de probabilidad (FDP).

2.1. Esperanza matemática y varianza

La esperanza matemática es igual al sumatorio de las probabilidades de que exista un suceso aleatorio, multiplicado por el valor del suceso aleatorio. Se puede calcular de la siguiente manera:

$$E(X) = \mu_x = \sum_{i=1}^{k} x_i \,.\, p_x(x_i) \tag{2.1.1}$$

La varianza es una medida de dispersión que representa la variabilidad de una serie de datos respecto a su media. Se puede calcular de la siguiente manera:

$$V(X) = \sigma_X^2 = \sum_{i=1}^{k} (x_i - \mu_x)^2 \,.\, p_x(x_i) \tag{2.1.2}$$

2.2. Función de distribución acumulada (FDA)

Es una función matemática que depende de una variable aleatoria real y de una distribución de probabilidad determinada que devuelve la probabilidad de que la variable sea igual o menor que un valor concreto.

$$F(x) = P(X \leq x) \tag{2.2.1}$$

Función de distribución acumulada inversa es la función inversa de la Función de distribución acumulada.

2.3. Función de densidad de probabilidad (FDP)

Describe la probabilidad relativa según la cual dicha v.a tomará determinado valor. La probabilidad de que la variable aleatoria caiga en una región específica del espacio de posibilidades estará dada por la integral de la densidad de esta variable entre uno y otro límite de dicha región.

2.4. Teorema Central del Limite (TCL)

Si $X_1, X_2, ..., X_n$ son n variables aleatorias independientes cualesquiera, con $E(X_i) = \mu_i$ y $V(X_i) = \sigma^2$ entonces, bajo ciertas condiciones que son válidas en la mayor parte de las aplicaciones, la variable aleatoria $X = X_1 + X_2 + ... + X_n$ con n convenientemente grande, tiene una distribución aproximadamente normal con parámetros:

$$E(X) = \sum_{i=1}^{n} \mu_i \qquad y \qquad V(X) = \sum_{i=1}^{n} \sigma_i^2 \tag{2.4.1}$$

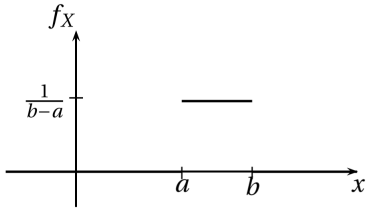
3. Distribuciones

3.1. Distribución de Uniforme continua

Decimos que una variable aleatoria X se distribuye uniformemente en el intervalo $[a, b]$ y notamos $X \sim U(a, b)$, cuando su función de densidad de probabilidad es

$$f_X(x) = \begin{cases} \frac{1}{b-a} & \text{si } x \in [a, b] \\ 0 & \text{en otro caso} \end{cases} \tag{3.1.1}$$

El hecho de que una variable aleatoria tenga un comportamiento uniforma significa que intervalos de igual amplitud contenidos en el intervalo $[a, b]$ tienen la misma probabilidad de ocurrir.



3.1.1. Esperanza matemática y varianza

Sea $X \sim U(a, b)$. Entonces

$$E(X) = \mu_X = \int_a^b \frac{x}{b-a} dx = \frac{a+b}{2} \tag{3.1.2}$$

$$V(X) = \sigma_X^2 = \int_a^b (x - \frac{a+b}{2})^2 \cdot \frac{1}{b-a} dx = \frac{(b-a)^2}{12} \tag{3.1.3}$$

Con esto, los parámetros pueden ser calculados

$$a = \mu_X - \sqrt{3\sigma_X^2} \tag{3.1.4}$$

$$b = 2\mu_X - a \tag{3.1.5}$$

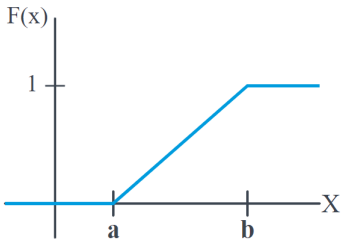
3.1.2. Distribución acumulada

La función de distribución acumulada de una variable aleatoria uniforme X es:

$$F(x) = \int_a^x \frac{x-a}{b-a} dw = \left[\frac{w}{b-a} \right]_a^x = \frac{x-a}{b-a} \tag{3.1.6}$$

entonces

$$f_X(x) = \begin{cases} 0 & x \leq a \\ \frac{x-a}{b-a} & a < x < b \\ 1 & x \geq b \end{cases} \tag{3.1.7}$$



3.1.3. Generación de números aleatorios

Dado que contamos con un conjunto de números aleatorios o pseudoaleatorios, que pertenecen al rango $[0, 1]$ y están distribuidos uniformemente, y que toda función de distribución acumulada arroja resultados en el mismo rango y con la misma distribución, puede pensarse a una variable aleatoria que sigue determinada distribución, como el conjunto de resultados de la función inversa de distribución acumulada.

En el caso en que se busca una v.a. distribuida uniformemente, partiendo de la función de la distribución acumulada F(x):

$$r = F(X) = \int_a^x \frac{1}{b-a} dt = \frac{x-a}{b-a} \qquad 0 \leq F(X) \leq 1 \tag{3.1.8}$$

$$r = \frac{x-a}{b-a} \tag{3.1.9}$$

$$x - a = r(b-a) \tag{3.1.10}$$

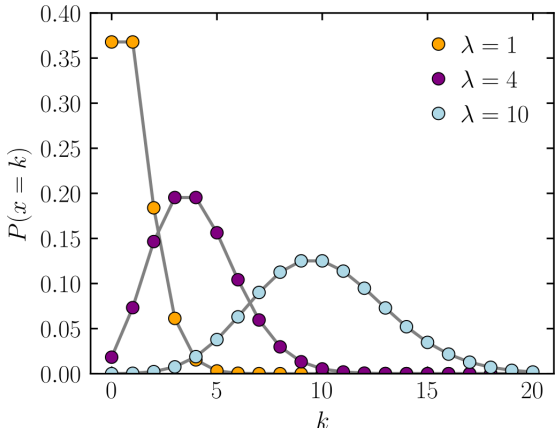
Llegando finalmente a:

$$x = r(b-a) + a \qquad 0 \leq r \leq 1 \tag{3.1.11}$$

3.2. Distribución de Poisson

La distribución de Poisson es una distribución de probabilidad discreta que expresa, a partir de una frecuencia de ocurrencia media λ, la probabilidad que ocurra un determinado número de eventos $k \in \mathbf{X}$ durante un intervalo de tiempo dado o una región específica.
Sea **X** una variable aleatoria que representa el número de eventos aleatorios independientes que ocurren a una rapidez constante sobre el tiempo o el espacio. Se dice entonces que la variable aleatoria **X** tiene una distribución de Poisson con parámetro $\lambda > 0$ y notamos $X \sim P(\lambda)$ y con una función de densidad de probabilidad.

$$f_X(k) = \begin{cases} \frac{\lambda^k e^{-\lambda}}{k!} & \text{si } x \geq 0 \\ 0 & \text{en otro caso} \end{cases} \tag{3.2.1}$$



En consecuencia , la función de distribución acumulada será:

$$F(X) = P(X \leq k) = \sum_{k=1}^{\infty} \frac{\lambda^k e^{-\lambda}}{k!} \tag{3.2.2}$$

3.2.1. Esperanza matemática y varianza

$$E(X) = V(X) = \lambda \tag{3.2.3}$$

3.2.2. Generación de números aleatorios

Para simular una distribución de Poisson con parámetro λ hacemos uso de la relación conocida entre las distribuciones Exponenciales y de Poisson.
Para generar valores con esta distribución, se generan primero duraciones de intervalos con distribución exponencial con un parámetro λ = 1 hasta que la suma de estos alcance o exceda el valor de λ.
En términos matemáticos, el valor de Poisson x se determina haciendo uso de la siguiente desigualdad:

$$\sum_{i=0}^{x} t_i \leq \lambda \leq \sum_{i=0}^{x+1} t_i \tag{3.2.4}$$

donde los valores de la variable t_i se generan por medio de la fórmula $t_i = -log(r_i)$ entonces:

$$\sum_{i=0}^{x} -log(r_i) \leq \lambda \leq \sum_{i=0}^{x+1} -log(r_i) \quad \Rightarrow \quad -log[\sum_{i=0}^{x}(r_i)] \leq \lambda \leq -log[\sum_{i=0}^{x+1}(r_i)] \quad \Rightarrow \quad log[\sum_{i=0}^{x}(r_i)] \geq \lambda \geq log[\sum_{i=0}^{x+1}(r_i)]$$

y por último:

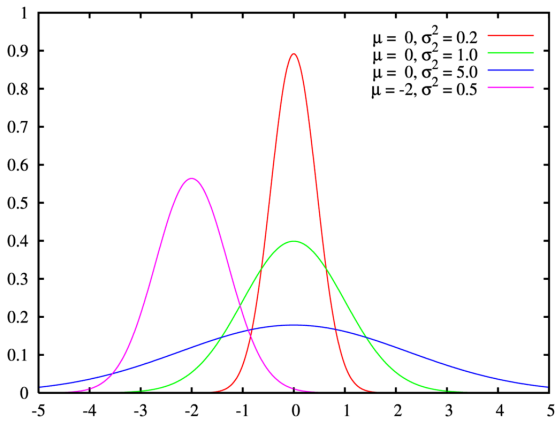
$$\sum_{i=0}^{x}(r_i) \geq e^{-\lambda} \geq \sum_{i=0}^{x+1}(r_i) \tag{3.2.5}$$

De esta manera, para generar nuestros números aleatorios con distribución de Poisson multiplicamos tantos r_i distribuidos uniformemente hasta que sea menor que $e^{-\lambda}$, aumentando el posible valor Poisson en 1 cada vez.

3.3. Distribución Normal

Decimos que una variable aleatoria X tiene una distribución normal de parámetros μ y σ, donde $\sigma > 0$, y notamos $X \sim N(\mu, \sigma)$ cuando su función de densidad de probabilidad es

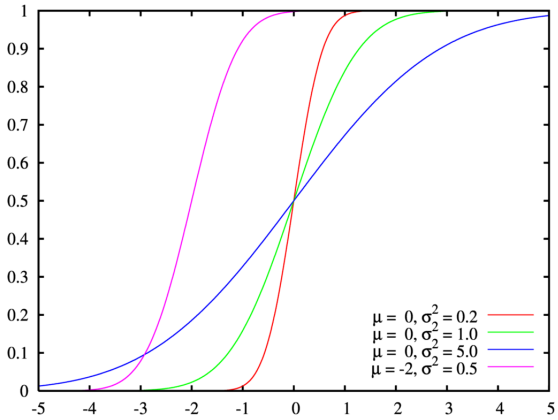
$$f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{3.3.1}$$



3.3.1. Distribución acumulada

La función de distribución acumulada de una variable aleatoria normal X es:

$$F(X) = P(X \leq k) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-(x-\mu)^2}{2\sigma^2}} dx \tag{3.3.2}$$



3.3.2. Esperanza matemática y varianza

Se demuestra que si $X \sim N(\mu, \sigma)$, entonces

$$E(X) = \int_{-\infty}^{\infty} x f_x(x)\, dx = \mu \tag{3.3.3}$$

$$V(X) = \int_{-\infty}^{\infty} (x-\mu)^2 f_x(x)\, dx = \sigma^2 \tag{3.3.4}$$

3.4. Distribución Normal Estándar

Si los parámetros $\mu_x = 0$ y $\sigma_x = 1$ la función de distribución recibirá el nombre de distribución normal estándar, con la función densidad:

$$f(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}z^2} \tag{3.4.1}$$

Cualquier distribución normal se puede convertir en estándar mediante la sustitución:

$$z = \frac{x - \mu_x}{\sigma_x} \tag{3.4.2}$$

3.4.1. Generación de números aleatorios

Para simular una distribución normal con media μ_x y varianza σ_x^2 se debe utilizar el **Teorema Central del Límite (TCL)**, el cual establece que la suma de n variables aleatorias independientes se aproxima a una distribución normal a medida que n se aproxima al infinito.

Si $r_1, r_2, ..., r_i$ representan v.a independientes donde cada una posee la misma distribución de probabilidad caracterizada por $E(r_i) = \theta$ y $V(r_i) = \sigma^2$, entonces:

$$\lim_{N->\infty} P\left[a < \frac{\sum_{i=1}^{N} r_i - N\theta}{\sqrt{N}\sigma} < b\right] = \frac{1}{\sqrt{2\pi}} \int_a^b e^{-1/2z^2} dz \tag{3.4.3}$$

Donde:

$$E(\sum_{i=1}^{N} r_i) = N\theta \tag{3.4.4}$$

$$V(\sum_{i=1}^{N} r_i) = N\theta^2 \tag{3.4.5}$$

$$z = \frac{\sum_{i=1}^{N} r_i - N\theta}{\sigma\sqrt{N}} \tag{3.4.6}$$

El procedimiento para simular valores normales utilizando computadoras requiere el uso de la suma de K valores de v.a distribuidos uniformemente, esto es la suma de $r_1, r_2, ..., r_k$ con cada r_i definido dentro del intervalo $0 < r_i < 1$.

Aplicando la convención notacional de la forma matemática del **Teorema Central del Límite (TCL)** así como los conocimientos previos dela distribución uniforme, encontramos que:

$$\theta = \frac{a+b}{2} = \frac{0+1}{2} = \frac{1}{2} \tag{3.4.7}$$

$$\sigma = \frac{b-a}{\sqrt{12}} = \frac{1}{\sqrt{12}} \tag{3.4.8}$$

$$z = \frac{\sum_{i=1}^{K} r_i - \frac{K}{2}}{\sqrt{\frac{K}{12}}} \tag{3.4.9}$$

Por definición, z es un valor de variable aleatoria con distribución normal estándar que se puede escribir en la forma sugerida por la ecuación 3.4.2, donde x es un valor de variable aleatoria distribuido en forma normal que se va a simular, con media μ y varianza . Igualando las ecuaciones 3.4.9 y 3.4.2 obtendremos:

$$\frac{x - \mu}{\sigma} = z = \frac{\sum_{i=1}^{K} r_i - \frac{K}{2}}{\sqrt{\frac{K}{12}}} \tag{3.4.10}$$

y resolviendo para x, se tiene que:

$$x = \sigma(\frac{12}{K})^{\frac{1}{2}}(\sum_{i=1}^{K} r_i - \frac{K}{2}) + \mu \tag{3.4.11}$$

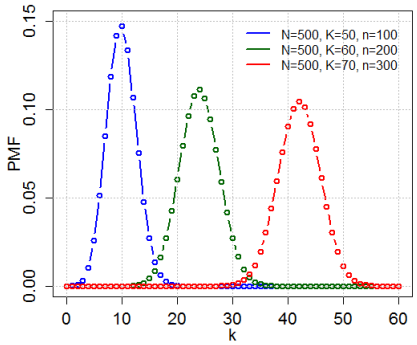
3.5. Distribución Hipergeométrica

Considerando una población de N elementos, tales que cada uno pertenece a la categoría A o la categoría B. Denotamos K al número de elementos que pertenecen a la categoría A y $N - K$ a los que pertenecen a la categoría B.

Si en una población de N elementos se toma una muestra aleatoria de $n < N$ sin lugar a reemplazo, el número de elementos x de la categoría A tendrá una distribución Hipergeométrica. La distribución está descrita por la siguiente función de probabilidad:

$$P(x) = \frac{\binom{K}{x}\binom{N-K}{n-x}}{\binom{N}{n}} \tag{3.5.1}$$

Donde $0 \leq x \leq K$ y $0 \leq n - x \leq N - K$.



3.5.1. Esperanza matemática y varianza

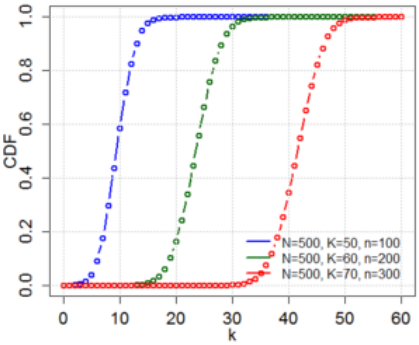
$$E(X) = \frac{nK}{N} \tag{3.5.2}$$

$$V(X) = \frac{nK}{N}(\frac{N-K}{N})(\frac{N-n}{N-1}) \tag{3.5.3}$$

3.5.2. Función acumulada

La función de distribución acumulada para la distribución hipergeométrica devuelve la probabilidad de que una observación, con una población de tamaño N, con R items, con muestra de tamaño n, y con ratio de posibilidades o, es menor o igual a x. Si o se omite o es igual a 1, se devuelve el valor de la distribución hipergeométrica regular.

$$CDF(x, N, R, n, o) = \begin{cases} 0 & x > \max(0, R+n-N) \\ \frac{\sum_{i=0}^{x}\binom{R}{i}\binom{N-R}{n-i}o^i}{\sum_{j=\max(0,R+n-N)}^{\min(R,n)}\binom{R}{j}\binom{N-R}{n-j}o^j} & \max(0, R+n-N) \leq x \leq \min(R, n) \\ 1 & \min(R, n) \end{cases} \tag{3.5.4}$$



3.5.3. Generación de números aleatorios

La generación de valores hipergeométricos involucra, substancialmente, la simulación de experimentos de muestreo sin reemplazo. En otras palabras, bastará sencillamente con que alteremos el método de ensayos de Bernoulli para generar valores binomiales, con objeto que N y p varíen en forma dependiente respecto al número total de elementos que previamente se han obtenido entre la población y el número de elementos de la clase I que se han extraído. A medida que se extrae un elemento de una muestra de n elementos, se reduce el valor de N = N_0 de acuerdo con la fórmula:

$$N_i = N_{i-1} - 1 \qquad i = 1, 2, ..., n \tag{3.5.5}$$

De manera similar, el valor de $p = p_0$ se transforma según

$$p_i = \frac{N_{i-1}p_{i-1} - S}{N_{i-1} - 1} \qquad i = 1, 2, ..., n. \tag{3.5.6}$$

a medida que se saca el i-ésimo elemento de la muestra de n elementos, donde $S = 1$ cuando el elemento de muestra $(i-1)$ pertenece a la clase I y S = 0 cuando el elemento de muestra $(i-1)$ pertenece a la clase II. Ciertamente, los valores iniciales de N_0 y P_0 corresponden a:

- N_0: tamaño inicial de la población.
- P_0: proporción de la población total que consta de elementos de la clase I.

Así, para una cantidad predefinida de iteraciones, podemos obtener una lista de longitud n de números distribuidos hipergeométricamente.

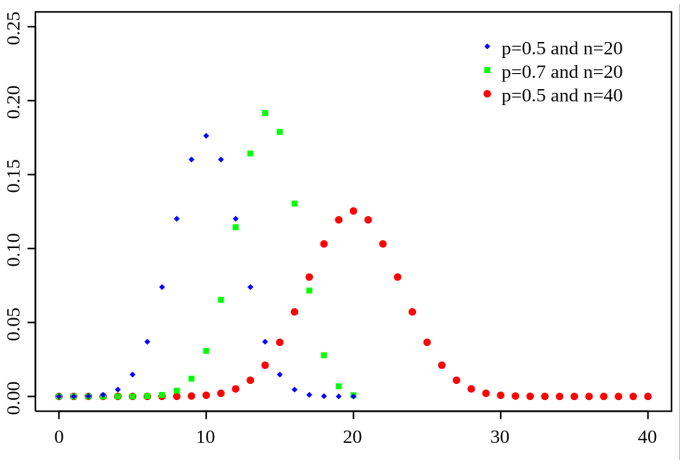
3.6. Distribución Binomial

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

Un experimento de Bernoulli se caracteriza por solo tener dos resultados son posibles, a uno de estos se le denomina "éxito" y tiene una probabilidad de ocurrencia p y al otro se le denomina "fracaso" y tiene una probabilidad $q = 1 - p$.

Definición: Sea A un suceso con probabilidad p. Entonces, la variable aleatoria Y : 'cantidad de veces que se presenta el suceso A en n repeticiones independientes' tiene un comportamiento o distribución binomial con parámetros n y p. En símbolos $Y \sim B(n, p)$

$$p_X(k) = P(Y = k) = \binom{n}{k} p^k (1-p)^{n-k} \qquad para \quad k = 0, 1, 2, 3...n \tag{3.6.1}$$



La función de distribución acumulada de una variable aleatoria $X \sim Bin(n, p)$ está dada por

$$F_X(X) = P(x \leq x) = \sum_{k=0}^{x} \binom{n}{k} p^k (1-p)^{n-k} \tag{3.6.2}$$

3.6.1. Esperanza matemática y varianza

$$E(X) = n \cdot p \tag{3.6.3}$$

$$V(X) = n \cdot p(1-p) \tag{3.6.4}$$

3.6.2. Generación de números aleatorios

Los valores de variables aleatorias con distribución Binomial se pueden generar de diversos modos, en el caso en el que el valor de n es moderado resulta que el método más eficiente es el basado en la reproducción de ensayos de Bernoulli siguiendo el método de rechazo.
Con este método se fija $x_0 = 0$, y conociendo los valores de p y de n, se generan n números aleatorios y para cada uno de ellos se efectúa una prueba. La variable x_i se incrementa de acuerdo con el siguiente criterio:

$$x_i = x_{i-1} + 1 \qquad si \quad r_i \leq p \tag{3.6.5}$$

$$x_i = x_{i-1} \qquad si \quad r_i > p \tag{3.6.6}$$

Después de haberse generado n números aleatorios, el valor de x_n será igual al valor de la variable aleatoria con distribución binomial x. Este procedimiento se puede repetir tantas veces como valores binomiales se requieran.

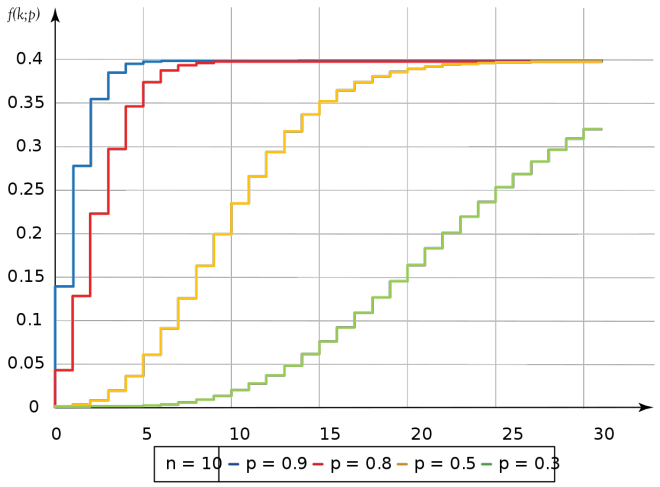
3.7. Distribución Pascal

La distribución de Pascal, también llamada distribución Binomial negativa, devuelve la probabilidad de $r-1$ éxitos y x fracasos en $x+r-1$ intentos, con éxito en el intento $(x+r)$, en una secuencia de p intentos independientes de Bernoulli. La función de densidad de probabilidad está dada por:

$$P_{r,p}(x) = p\left[\binom{x+r-1}{x1} p^{r-1}(1-p)^x\right]p \tag{3.7.1}$$

$$P_{x,r,p} = \binom{x+r-1}{r-1} p^r (1-p)^x \tag{3.7.2}$$

donde r es la cantidad de éxitos ($\mathbb{Z}$) y $\binom{n}{k}$ es un coeficiente binomial.



3.7.1. Esperanza matemática y varianza

$$E(X) = \frac{rq}{p} \tag{3.7.3}$$

$$V(X) = \frac{rq}{p^2} \tag{3.7.4}$$

3.7.2. Función de distribución acumulada

La función acumulada de una distribución binomial negativa es

$$CDF(x, p, r) = \begin{cases} 0 & x < 0 \\ p^r \sum_{j=0}^{x} \binom{r+j-1}{r-1} (1-p)^j & x \geq 0 \end{cases} \tag{3.7.5}$$

Puede ser expresada en términos del a distribución acumulada binomial:

$$F(x, r, p) = F_{binomial}(x; n = x + r, p) \tag{3.7.6}$$

donde los parámetros se pueden calcular como

$$p = \frac{E(X)}{V(X)} \tag{3.7.7}$$

$$r = \frac{E(X)^2}{V(X) - E(X)} \tag{3.7.8}$$

3.7.3. Generación de números aleatorios

Para simular una distribución de Pascal deberemos tener en cuenta el siguiente procedimiento:
Para una media y varianza dadas, se pueden determinar los parámetros p y k de la siguiente manera:

$$p = \frac{E(X}{V(X)} \tag{3.7.9}$$

$$k = \frac{E(X)^2}{V(X) - E(X)} \tag{3.7.10}$$

Sin embargo, puede suceder que el proceso de simulación se complique considerablemente cuando resulte que en la ecuación anterior el valor que se obtenga al efectuar el cómputo de k no sea entero. Cuando k es un entero, los valores de la variable aleatoria con distribución de Pascal se pueden generar con sólo considerar la suma de k valores con distribución geométrica (k = 1). En consecuencia:

$$x = \frac{\sum_{i=1}^{k} (ln\ r_i)}{ln\ q} \Rightarrow x = \frac{ln(\sum_{i=1}^{k} r_i)}{ln\ q} \tag{3.7.11}$$

Viene a ser un valor de variable aleatoria con distribución de Pascal, una vez que su magnitud se redondea con respecto al menor entero más próximo al valor calculado.

Así, para una cantidad predefinida de iteraciones, podemos obtener una lista de longitud n de números distribuidos como binomial negativa.

3.8. Distribución Gamma

Es una distribución de probabilidad continua adecuada para modelizar el comportamiento de variables aleatorias con asimetría positiva y/o los experimentos en donde está involucrado el tiempo.
Una variable aleatoria tiene una distribución gamma si su función de densidad está dada por:

$$f_X(x) = \begin{cases} \frac{\lambda(\lambda x)^{\alpha-1}e^{\lambda x}}{\Gamma(\alpha)} & \text{Para } x > 0, \alpha > 0, \lambda > 0 \\ 0 & \text{en otro caso} \end{cases} \tag{3.8.1}$$

La función de distribución acumulada de una variable aleatoria $X \sim \Gamma(\alpha, \lambda)$ está dada por

$$F(X) = \int_0^x \frac{\lambda(\lambda y)^{\alpha-1}e^{\lambda y}}{\Gamma(\alpha)} dy \tag{3.8.2}$$

Si X es una variable aleatoria tal que $X \sim \Gamma(n, \lambda)$ donde $n \in Z$ (es decir, X tiene una distribución de Erlang) entonces su función de distribución acumulada está dada por

$$F(X) = \sum_{k=n}^{\infty} \frac{(\lambda x)^k e^{-\lambda x}}{k!} \tag{3.8.3}$$

3.8.1. Esperanza matemática y varianza

$$E(X) = \frac{\alpha}{\lambda} \tag{3.8.4}$$

$$V(X) = \frac{\alpha}{\lambda^2} \tag{3.8.5}$$

3.8.2. Generación de números aleatorios

Para generar valores de la v.a de la distribución gamma se pueden emplear las siguientes fórmulas a fin de determinar los parámetros de la función densidad de probabilidad Gamma:

$$\alpha = \frac{E(X)}{V(X)} \tag{3.8.6}$$

$$K = \frac{E(X)^2}{V(X)} \tag{3.8.7}$$

Debido a que la función distribución gamma no se puede formular explícitamente una función de distribución acumulativa, no podemos hacer uso de la función inversa para la generación de números aleatorios. Debemos considerar un método alternativo, para ello se hace uso del mismo proceso aleatorio sobre el cual se basa la distribución de Erlang. Se toma los k valores de la v.a con distribución exponencial $x_1, x_2, ..., x_k$ cuyo valor esperado es el mismo e igual a $1/\lambda$. En consecuencia, el valor de la v.a (Erlang) x se puede expresar como:

$$x = \sum_{i=1}^{k} x_i = -\frac{1}{\alpha} \sum_{i=1}^{k} log(r_i) \tag{3.8.8}$$

$$x = -\frac{1}{\alpha} log(\sum_{i=1}^{k} r_i) \tag{3.8.9}$$

Nótese que en esta implementación, solo se consideran los valores en los cuales k es un entero positivo.

El problema de generar valores de variable aleatoria, cuando el parámetro k no es un entero, queda aún en la actividad como un problema por resolverse, ya que en este caso todavía se formula un método estocástico satisfactorio, sin lo cual no puede justificarse algún proceso de simulación correspondiente.

3.9. Distribución de Erlang[6]

Un caso particular de distribución Gamma es cuando $X \sim \gamma(\alpha, \lambda)$ con $\alpha \in N$ y a esto se lo conoce como distribución de Erlang.
La distribución de Erlang es una distribución de continua, con parámetros n (factor de forma de la distribución) y λ (factor de proporción de la distribución).
Una variable aleatoria continua X tiene distribución de Erlang con parámetros $n \in Z^+$ y $\lambda > 0$ [en símbolos $X \sim Erlang(n, \lambda)$] si su función de densidad para valores de x mayores a cero es:

$$f(x) = \frac{\lambda(\lambda x)^{n-1} e^{-\lambda x}}{(n-1)!} \tag{3.9.1}$$

Propiedades:

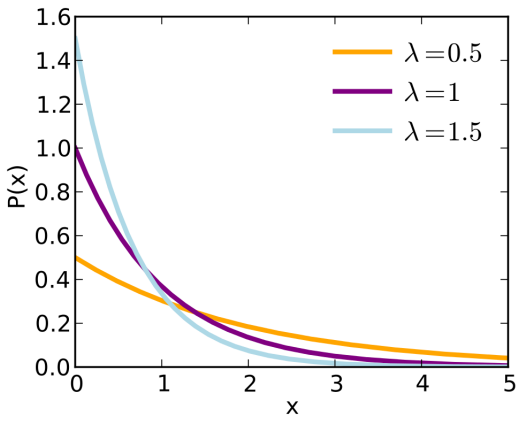
- Esperanza matemática: $E(X) = \frac{n}{\lambda}$
- Varianza: $Var(X) = \frac{n}{\lambda^2}$
- Función generadora de momentos: $M_X(t) = (\frac{\lambda}{\lambda - t})^n$

Así como la distribución de Erlang es un caso particular de distribución Gamma, al caso particular de $Erlang(n, \lambda)$ donde $Erlang(1, \lambda)$ se lo conoce como distribución Exponencial.

3.10. Distribución Exponencial

La distribución exponencial es una distribución continua que se utiliza para modelar tiempos de espera para la ocurrencia de un cierto evento.
Función de densidad Se dice que una variable aleatoria continua X tiene una distribución exponencial con parámetro $\lambda > 0$ y escribimos $X \sim Exp(\lambda)$ si su función de densidad de probabilidad es

$$f_X(x) = \begin{cases} \lambda e^{-\lambda x} & \text{si } x \geq 0 \\ 0 & \text{en otro caso} \end{cases} \tag{3.10.1}$$



En consecuencia , la función de distribución acumulada será:

$$F(X) = \int_0^x \lambda e^{-\lambda x} dx = 1 - e^{-\lambda x} \tag{3.10.2}$$

3.10.1. Esperanza matemática y varianza

$$E(X) = \frac{1}{\lambda} \tag{3.10.3}$$

$$V(X) = \frac{1}{\lambda^2} \tag{3.10.4}$$

3.10.2. Generación de números aleatorios

Debido a la simetría que existe entre la distribución uniforme, sigue que la intercambiabilidad de $F(x)$ y $1F(x)$. Por lo tanto:

$$r = e^{-\lambda x} \tag{3.10.5}$$

y consecuentemente:

$$x = -(\frac{1}{\lambda}) \log(r) = -\mu \, log(r) \qquad 0 \leq r \leq 1 \tag{3.10.6}$$

3.11. Distribución Empírica discreta [11]

Las distribuciones empíricas sirven para datos que siguen un orden numérico natural.
Sea $h(x_j)$ la frecuencia absoluta de observaciones de una V.A. discreta, al numero de observaciones que no superan ese valor se lo denomina frecuencia absoluta acumulada $H(x_j)$:

$$H(x_j) = \sum_{s=1}^{j} h(x_s) \text{con } j = 1, 2, ..., k \tag{3.11.1}$$

La frecuencia relativa acumulada es:

$$F(x_j) = \frac{H(x_j)}{n} = \sum_{s=1}^{j} f(x_s) \tag{3.11.2}$$

4. Métodos para la Generacion de numeros aleatorios

A continuación se explicaran métodos para obtener las distribuciones explicadas previamente a partir de números pseudoaleatorios generados uniformemente entre 0 y 1 ($U \sim (0, 1)$). Se sugiere usar el siguiente generador de numeros pseudo aleatorios para probar las funciones:

```
import numpy as np

def generador_numpy(n):
    numbers = []
    for i in range(n):
        numbers.append(np.random.uniform(0, 1))
    return numbers
```

4.1. Método de transformación inversa

Para generar valores x_i de una v.a. que siga una función de densidad $f(x)$ se necesita de la función de distribución acumulativa $F(x)$ ya que su rango es $[0, 1]$ y podemos generar números aleatorios distribuidos uniformemente en este rango. Puesto que $F(x) = r$ queda claro que:

$$x = F^{-1}(r) \tag{4.1.1}$$

Así cualquier valor r_0 que generemos nos permite encontrar un valor x_0 que se corresponda con el. Generada una v.a distribuida uniformemente en $[0, 1]$ correspondientes a $F(x)$ se puede resumir el método a:

$$r = F(x) = \int_{-\infty}^{x} f(t)\, dt \tag{4.1.2}$$

y entonces

$$P(X \leq x) = F(x) = P[r \leq F(x)] = P[F^{-1}(r) \leq x] \tag{4.1.3}$$

En muchas distribuciones de probabilidad resulta imposible o muy difícil expresar x como la transformación inversa $F^{-1}(r)$. Para esto será necesario recurrir a otros métodos.

4.1.1. Generación de una distribución uniforme con método transformación inversa

La función de densidad de una distribución uniforme entre a y b es:

$$f_X(x) = \begin{cases} \frac{1}{b-a} & \text{si } a \leq x \leq b \\ 0 & \text{en otro caso} \end{cases} \tag{4.1.4}$$

Y su función de densidad acumulada es:

$$F(x) = \int_{-\infty}^{x} f_X(x)dx = \int_{a}^{x} f_X(x)dx = \int_{a}^{x} \frac{1}{b-a}dx = \frac{x-a}{b-a} \tag{4.1.5}$$

Dado un $r = F(x)$ entonces:

$$r = \frac{x-a}{b-a} \tag{4.1.6}$$

$$r(b-a) = x - a$$

$$x = a + (b-a)r \tag{4.1.7}$$

Codigo Python 3 para generar la distribución a partir de este método

```
def UniformeT(pseudo: list, a: float, b: float) -> list:
    """pseudo: Lista de numeros pseudoaleatorios
    a:valor minimo
    b:valor maximo
    returns: Lista distribuida uniformemente en [a,b]"""
    # la func de densidad es 1/b-a
    # la func de acumulacion es x-a/b-a
    uni = []
    for r in pseudo:
        x = a+(b-a)*r
        uni.append(x)
    return uni
```

4.1.2. Generación de una distribución exponencial con método transformación inversa

La función de densidad de una distribución exponencial con parámetro λ es:

$$f_X(x) = \begin{cases} \lambda e^{-\lambda x} & \text{si } x \geq 0 \\ 0 & \text{en otro caso} \end{cases} \tag{4.1.8}$$

Y su función de densidad acumulada es:

$$F(x) = \int_{-\infty}^{x} f_X(x)dx = \int_{0}^{x} \lambda e^{-\lambda x}dx = 1 - e^{-\lambda x} \tag{4.1.9}$$

Dado un $r = F(x)$ entonces:

$$r = 1 - e^{-\lambda x} \tag{4.1.10}$$

$$1 - r = e^{-\lambda x}$$

$$\ln(1 - r) = \ln(e^{-\lambda x})$$

$$-\lambda x = \ln(1 - r)$$

$$x = \frac{\ln(1 - r)}{-\lambda} \tag{4.1.11}$$

Codigo Python 3 para generar la distribución a partir de este método

```
from math import log


def ExponencialT(pseudo: list, lmbda: float) -> list:
    expo = []
    for r in pseudo:
        x = log(1-r)/(-lmbda)
        expo.append(x)
    return expo
```

4.1.3. Generación de una distribución normal con método transformación inversa [9][10]

La función de densidad de una distribución normal con parámetros λ y α:

$$f_X(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{4.1.12}$$

Y su función de densidad acumulada es:

$$F(X) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-(x-\mu)^2}{2\sigma^2}} dx \tag{4.1.13}$$

Dado un $r = F(x)$ entonces:

$$r = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-(x-\mu)^2}{2\sigma^2}} dx \tag{4.1.14}$$

$$r = \frac{1}{2}(1 + erf(\frac{x-\mu}{\sqrt{2}\sigma})) \qquad \text{Donde } \textbf{erf} \text{ es la función error 8} \tag{4.1.15}$$

$$2r - 1 = erf(\frac{x-\mu}{\sqrt{2}\sigma}) \tag{4.1.16}$$

Dado un número complejo z, no existe un único numero complejo w tal que $erf(w) = z$, pero para $-1 \leq y \leq 1$ existe un único real que satisface $y = erf^{-1}(erf(y))$. Dado que r se encuentra entre 0 y 1, $0 \leq 2r - 1 \leq 1$ con lo cual pude hacerse la inversa de la función error para $\frac{x-\mu}{\sqrt{2}\sigma}$:

$$erf^{-1}(2r-1) = \frac{x-\mu}{\sigma\sqrt{2}} \tag{4.1.17}$$

$$x - \mu = erf^{-1}(2r-1)\sigma\sqrt{2} \tag{4.1.18}$$

$$x = \mu + erf^{-1}(2r-1)\sigma\sqrt{2} \tag{4.1.19}$$

Codigo Python 3 para generar la distribución a partir de este método

```
from math import sqrt
from scipy.special import erfinv

def NormalT(pseudo: list, mu: float, sigma: float) -> list:
    norm = []
    for r in pseudo:
        x = mu+(sqrt(2)*sigma*erfinv((2*r)-1))
        norm.append(x)
    return norm
```

4.2. Método de rechazo

El método de rechazo, también conocido como método de aceptación-rechazo o método de Montecarlo, es una técnica utilizada para generar números aleatorios que siguen una determinada distribución de probabilidad. Este método se basa en la idea de generar números aleatorios a partir de una distribución de probabilidad más simple, y luego se seleccionan solo aquellos que cumplen ciertas condiciones para que se ajusten a la distribución objetivo.
Si $f(x)$ es una función de probabilidad acotada y $x \in [a, b]$ se puede usar la siguiente técnica llamada técnica de rechazos para generar valores de variables aleatorias:

1. Normalizar el rango de f con un factor de escala c tal que:

$$cf(x) \leq 1 \tag{4.2.1}$$

2. Definir a x como una función lineal de r

$$x = a + (b-a)r \tag{4.2.2}$$

3. Generar parejas de números aleatorios $(r_1 r_2)$ donde estos satisfacen:

$$r_2 \leq cf[a + (b-a)r_1] \tag{4.2.3}$$

Como este par sera aceptado se usa $x = a + (b-a)r_1$ para generar valores de la v.a.
Este método se basa en el hecho de la probabilidad de que $r \leq c\,.\,f(x)$ es:

$$P[r \leq c\,.\,f(x)] = c\,.\,f(x) \tag{4.2.4}$$

Por consiguiente, si un número es escogido al azar de acuerdo a $x = a + (b-a)r_1$ y si $r > c * f(x)$ entonces se rechaza y la distribución de probabilidad de las x aceptadas será exactamente $f(x)$. Por otra parte, conviene señalar que si todas las x's fueran aceptadas, entonces x estaría uniformemente distribuida entre a y b.

A continuacion se mostrara como generar V.A. que siguen determinadas distribuciones, usando este método, con código de Python 3.

4.2.1. Distribución uniforme- Método de rechazo

```
import numpy as np

def UniformeR(pseudo: list, a: float, b: float) -> list:
    aceptados = []
    M = 1/(b-a)
    for U in pseudo:
        while True:
            V = np.random.uniform(0, 1)
            T = a+((b-a)*V)
            if(M*U <= 1/(b-a)):
                aceptados.append(T)
                break
    return aceptados
```

4.2.2. Distribuciones Gamma y Exponencial- Método de rechazo

```
import numpy as np
from math import sqrt, pi, exp, gamma

def densidad_gamma(lmbda: float, alpha: float, x: float) -> float:
    numerador = lmbda*((lmbda*x)**(alpha-1))*exp(-lmbda*x)
    denominador = gamma(alpha)
    return numerador/denominador


def GammaR(pseudo: list, lmbda: float, alpha: float, max_x: float) -> list:
    x_max = (alpha-1)/lmbda  # x en el que se da la maxima densidad
    M = densidad_gamma(lmbda=lmbda, alpha=alpha, x=x_max)
    a, b = 0, max_x
    gmma = []
    for U in pseudo:
        while True:
            V = np.random.uniform(0, 1)
            T = b*V  # a=0 siempre

            if(M*U <= densidad_gamma(lmbda=lmbda, alpha=alpha, x=T)):
                gmma.append(T)
                break
    return gmma


def ExponenecialR(pseudo: list, lmbda: float, max_x: float) -> list:
    """Una distribucion exponencial es un caso especial de
    distribucion Gamma donde el parametro de forma alfa=1"""
    return GammaR(pseudo=pseudo, lmbda=lmbda, alpha=1, max_x=max_x)
```

4.2.3. Distribución Normal- Método de rechazo

```
import numpy as np
from math import sqrt, pi, exp

def densidad_norm(m, v, x):
    numerador = exp(-(((x-m)**2)/2*(v**2)))
    denominador = v*sqrt(2*pi)
    return numerador/denominador


def NormalR(pseudo: list, mu: float, des: float) -> list:
    """Aproximacion a una distribucion N\sim (mu,des) con metodo
    de aceptacion y rechazo de Von Neumann
    pseudo: Elementos pseudoaleatorios con distribucion uniforme
    mu:media de la funcion objetivo
```

```
    des:desvio de la funcion objetivo"""
    a, b = mu-(10*des), mu+(10*des)
    M = 1/(des*sqrt(2*pi))  # maximo valor de densidad de la funcion objetivo
    aceptados = []
    for U in pseudo:
        while True:
            V = np.random.uniform(0, 1)
            T = a+((b-a)*V)
            if(M*U <= densidad_norm(mu, des, T)):
                # T se acepta
                aceptados.append(T)
                break
    norm = [T for T in aceptados]
    return norm
```

4.2.4. Distribución Binomial- Método de rechazo

```
import numpy as np
from math import exp, factorial, trunc

def densidad_binomial(n: int, p: float, x: int) -> float:
    a = factorial(n)/(factorial(x)*(factorial(n-x)))
    b = p**x
    c = (1-p)**(n-x)
    return a*b*c


def BinomialR(pseudo: list, n: int, p: float) -> list:
    M = densidad_binomial(n, p, trunc(n*p))
    binom = []
    for U in pseudo:
        while True:
            V = np.random.uniform(0, 1)
            T = trunc(n*V)
            if(M*U <= densidad_binomial(n, p, T)):
                binom.append(T)
                break
    return binom
```

4.2.5. Distribución de Pascal- Método de rechazo

```
import numpy as np
from math import sqrt, pi, exp, gamma, factorial, trunc

def densidad_Pascal(r: int, p: float, x: int) -> float:
    a = factorial(x+r-1)/(factorial(x)*(factorial(r-1)))
    b = p**r
    c = (1-p)**(x)
    return a*b*c


def PascalR(pseudo: list, r: int, p: float) -> list:
    M = densidad_Pascal(r, p, trunc(r*(1-p)/p))
    pascal = []
    for U in pseudo:
        while True:
            V = np.random.uniform(0, 1)
            T = trunc(2*r*V)
            if(M*U <= densidad_Pascal(r, p, T)):
                pascal.append(T)
                break
    return pascal
```

4.2.6. Distribución de Poisson- Método de rechazo

```
import numpy as np
from math import sqrt, pi, exp, gamma, factorial, trunc

def densidad_Poisson(lmbda: float, x: int) -> float:
    numerador = (lmbda**x)*exp(-lmbda)
    denominador = factorial(x)
    return numerador/denominador


def PoissonR(pseudo: list, lmbda: float, max_x: int) -> list:
    M = densidad_Poisson(lmbda=lmbda, x=trunc(lmbda))
    poisson = []
    for U in pseudo:
        while True:
            V = np.random.uniform(0, 1)
            T = trunc(V*max_x)  # a=0 siempre, b=max_x
            if(M*U <= densidad_Poisson(lmbda, T)):
                poisson.append(T)
                break
    return poisson
```

4.2.7. Distribución Empírica Discreta- Método de rechazo

```
import numpy as np
from math import sqrt, pi, exp, gamma, factorial, trunc

def EmpiricaR(pseudo: list, min_x: int, lista_fr: list) -> list:
    """pseudo: n pseudoaleatorio en [0,1]
    min_x:valor minimo que asumira la VA que sigue la distribucion empirica
    lista_fr:frecuencia relativa de cada numero consecutivo a partir de min_x
    :!!! sum(lista_fr)==1 !!!"""
    n = len(lista_fr)
    a, b = min_x, min_x+n
    frec_rel = dict()
    M = max(lista_fr)
    empiric = []
    for i in range(a, b):
        frec_rel.setdefault(i, lista_fr.pop(0))
    for U in pseudo:
        while True:
            V = np.random.uniform(0, 1)
            T = trunc(a+(b-a)*V)
            if T not in frec_rel.keys():
                break
            if(M*U <= frec_rel[T]):
                empiric.append(T)
                break
    return empiric
```

4.2.8. Distribución Hipergeométrica- Método de rechazo

```
import numpy as np
from math import factorial, trunc

def densidad_Hipergeometrica(N: int, K: int, n: int, x: int) -> float:
    a = factorial(K)/(factorial(x)*factorial(K-x))
    b = factorial(N-K)/(factorial(n-x)*factorial(N-K-(n-x)))
    c = factorial(N)/(factorial(n)*factorial(N-n))
    return (a*b)/c


def HipergeometricaR(pseudo: list, N: int, K: int, n: int) -> list:
```

```
    media = n*K/N
    M = densidad_Hipergeometrica(N, K, n, media)
    hiper = []
    for U in pseudo:
        while True:
            V = np.random.uniform(0, 1)
            T = trunc(n*V)
            if(M*U <= densidad_Hipergeometrica(N, K, n, T)):
                hiper.append(T)
                break
    return hiper
```

5. Código de distribuciones

El siguiente código usa los métodos definidos previamente(tanto método de transformación inversa como método de aceptación y rechazo) para generar las distribuciones, graficar su histograma(con matplotlib) y testear usando el test de Kolmogorov-Smirnov.
Todo el código previamente presentado y el que esta a continuación se encuentra en el siguiente repositorio: https://github.com/Faus14/Simulacion_2023/tree/main/TP-2.2-GeneradorDistribuido

```
from scipy.stats import uniform, norm, gamma, expon, binom, poisson, nbinom, geom
from matplotlib import pyplot as plt
import numpy as np
from math import sqrt


def generador_numpy(n):
    numbers = []
    for i in range(n):
        numbers.append(np.random.uniform(0, 1))
    return numbers


def KolmogorovTest(lista, alfa, dist_type):
    '''Test de Kolmogorov-Smirnov, compara el cdf(valor cr tico) de una distribucion
    con el cdf(d) de la muestra(lista) de tama o n, para el nivel de significancia alfa.
    Devuelde verdadero si la distribuci n es uniforme, falso si no lo es.'''

    lista.sort()  # Ordeno la lista de menor a mayor
    d_positivo = []  # array de los valores calculados para d positivo con la f rmula de KS
    d_negativo = []  # array de los valores calculados para d negativo con la f rmula de KS

    for i in range(len(lista)):
        # F rmula de KS para d positivo
        d_positivo.append(i / len(lista) - lista[i])
        # F rmula de KS para d negativo
        d_negativo.append(lista[i] - (i - 1) / len(lista))

    # Calculo el m ximo entre los d
    dmaximo = max(max(d_positivo), max(d_negativo))
    # Tomo el valor cr tico d de la tabla de KS
    if type(dist_type) is type(binom):
        k_tabla = binom.ppf(1 - alfa / 2, len(lista), p=p)
    elif type(dist_type) is type(nbinom):
        k_tabla = nbinom.ppf(1 - alfa / 2, len(lista), p=p)
    else:
        k_tabla = dist_type.ppf(1 - alfa / 2, len(lista))

    if dmaximo < k_tabla:  # Comparo el valor d de la muestra con el valor cr tico de la tabla
        # Hipotesis aceptada, distribucion es uniforme
        return f'Pasa prueba '
    # Hip tesis rechazada, distribucion no es uniforme
```

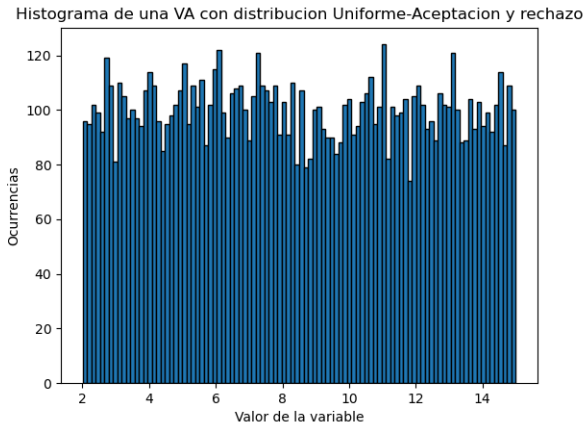
```
    return f'No pasa prueba '


cola = .05  # 1-cola/2 es el intervalo de confianza para el test Kolmogorov-Smirnov
a, b = 2, 15
dist = generador_numpy(10000)
dist = UniformeT(dist, a, b)
plt.hist(dist, bins=round(sqrt(len(dist))),  edgecolor='black')
plt.title('Histograma de una VA uniforme-Transformacion inversa')
plt.xlabel('Valor de la variable')
plt.ylabel('Ocurrencias')
plt.show()
print(f"Test Kolmogorov-Smirnov: Distribucion UniformeT entre {a=} y {b=}->
{KolmogorovTest(dist,cola,uniform)}")

mu, des = 2, .5
dist = generador_numpy(10000)
dist = NormalT(dist, mu, des)
plt.hist(dist, bins=round(sqrt(len(dist))),  edgecolor='black')
plt.title('Histograma de una VA con distribucion Normal- Transformacion inversa')
plt.xlabel('Valor de la variable')
plt.ylabel('Ocurrencias')
plt.show()
print(f"Test Kolmogorov-Smirnov: Distribucion NormalT {mu=} y {des=}->
{KolmogorovTest(dist,cola,norm)}")

lmbda_e = 2
dist = generador_numpy(10000)
dist = ExponencialT(dist, lmbda_e)
plt.hist(dist, bins=round(sqrt(len(dist))),  edgecolor='black')
plt.title('Histograma de una VA con distribucion Exponencial- Transformacion inversa')
plt.xlabel('Valor de la variable')
plt.ylabel('Ocurrencias')
plt.show()
print(f"Test Kolmogorov-Smirnov: Distribucion ExponencialT con {lmbda_e=}->
{KolmogorovTest(dist,cola,expon)}")


dist = generador_numpy(10000)
dist = UniformeR(dist, a, b)
plt.hist(dist, bins=round(
    sqrt(len(dist))),  edgecolor='black')
plt.title('Histograma de una VA con distribucion Uniforme-Aceptacion y rechazo')
plt.xlabel('Valor de la variable')
plt.ylabel('Ocurrencias')
plt.show()
print(f"Test Kolmogorov-Smirnov: Distribucion UniformeR entre {a=} y {b=}->
{KolmogorovTest(dist,cola,uniform)}")

dist = generador_numpy(10000)
dist = NormalR(dist, mu, des)
plt.hist(dist, bins=round(sqrt(len(dist))),  edgecolor='black')
plt.title('Histograma de una VA con distribucion Normal-Aceptacion y rechazo')
plt.xlabel('Valor de la variable')
plt.ylabel('Ocurrencias')
plt.show()
print(f"Test Kolmogorov-Smirnov: Distribucion NormalR {mu=} y {des=}->
{KolmogorovTest(dist,cola,norm)}")

lmbda_p, X_p = 10, 25
dist = generador_numpy(10000)
dist = PoissonR(dist, lmbda_p, X_p)
plt.hist(dist, bins=round(sqrt(len(dist))),  edgecolor='black')
plt.title('Histograma de una VA con distribucion de Poisson-Aceptacion y rechazo')
plt.xlabel('Valor de la variable')
```

```
plt.ylabel('Ocurrencias')
plt.show()
print(f"Test Kolmogorov-Smirnov: Distribucion PoissonR {lmbda_p=} y {X_p=}->
{KolmogorovTest(dist,cola,poisson)}")


n, p = 40, .5
dist = generador_numpy(10000)
dist = BinomialR(dist, n, p)
plt.hist(dist, bins=round(sqrt(len(dist))),  edgecolor='black')
plt.title('Histograma de una VA con distribucion Binomial-Aceptacion y rechazo')
plt.xlabel('Valor de la variable')
plt.ylabel('Ocurrencias')
plt.show()
print(f"Test Kolmogorov-Smirnov: Distribucion BinomialR {n=} y {p=}->
{KolmogorovTest(dist,cola,binom)}")

r = 80
dist = generador_numpy(10000)
dist = PascalR(dist, r, p)
plt.hist(dist, bins=round(sqrt(len(dist))),  edgecolor='black')
plt.title('Histograma de una VA con distribucion de Pascal-Aceptacion y rechazo')
plt.xlabel('Valor de la variable')
plt.ylabel('Ocurrencias')
plt.show()
print(f"Test Kolmogorov-Smirnov: Distribucion PascalR {r=} y {p=}->
{KolmogorovTest(dist,cola,nbinom)}")

lmbda_g, alpha_g, X_g = 2, 2, 20
dist = generador_numpy(10000)
dist = GammaR(dist, lmbda_g, alpha_g, X_g)
plt.hist(dist, bins=round(sqrt(len(dist))),  edgecolor='black')
plt.title('Histograma de una VA con distribucion Gamma-Aceptacion y rechazo')
plt.xlabel('Valor de la variable')
plt.ylabel('Ocurrencias')
plt.show()
print(f"Test Kolmogorov-Smirnov: Distribucion GammaR {lmbda_g=}, {alpha_g=} y {X_g=}->
{KolmogorovTest(dist,cola,gamma)}")

dist = generador_numpy(10000)
dist = ExponenecialR(dist, lmbda=lmbda_e, max_x=20)
plt.hist(dist, bins=round(sqrt(len(dist))),  edgecolor='black')
plt.title('Histograma de una VA con distribucion Exponencial-Aceptacion y rechazo')
plt.xlabel('Valor de la variable')
plt.ylabel('Ocurrencias')
plt.show()
print(f"Test Kolmogorov-Smirnov: Distribucion ExponencialR con {lmbda_e=}->
{KolmogorovTest(dist,cola,expon)}")

dist = generador_numpy(10000)
fr_empirica = [.01, .09, .03, .06, .04, .07, .1, .3, .3]
dist = EmpiricaR(dist, -9, fr_empirica)
plt.hist(dist, bins=round(sqrt(len(dist))),  edgecolor='black')
plt.title('Histograma de una VA con distribucion Empicrica-Aceptacion y rechazo')
plt.xlabel('Valor de la variable')
plt.ylabel('Ocurrencias')
plt.show()

N_g, K_g, n_g = 200, 60, 10
dist = generador_numpy(10000)
dist = HipergeometricaR(dist, N_g, K_g, n_g)
plt.hist(dist, bins=round(sqrt(len(dist))),  edgecolor='black')
plt.title(
    'Histograma de una VA con distribucion Hipergeometrica-Aceptacion y rechazo')
plt.xlabel('Valor de la variable')
```

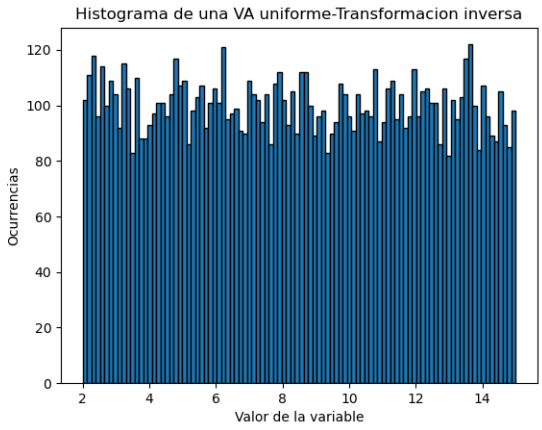
```
plt.ylabel('Ocurrencias')
plt.show()
print(
    f"Test Kolmogorov-Smirnov: Distribucion HiperGeometrica con {N_g=}, {K_g=} y {n_g=}->
    {KolmogorovTest(dist,cola,geom)}")
```

6. Histogramas de las distribuciones generadas

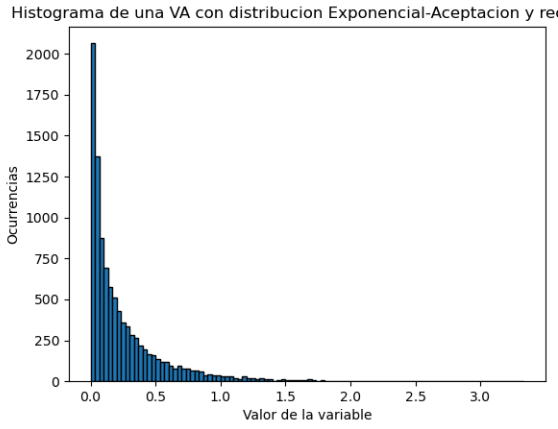
6.1. Distribución Uniforme- Aceptación/Rechazo



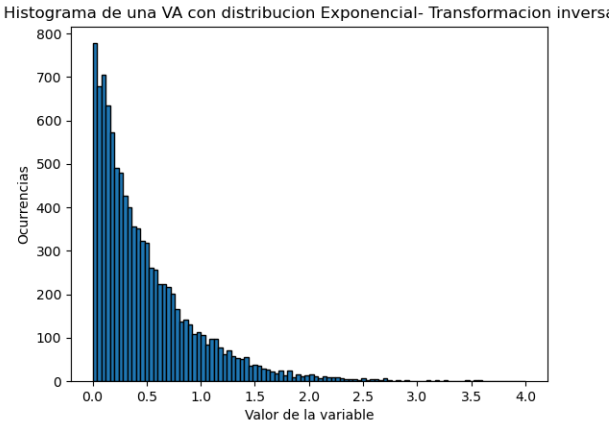
6.2. Distribución Uniforme- T. inversa



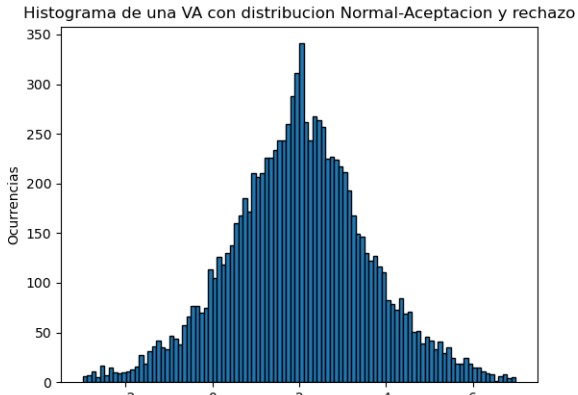
6.3. Distribución Exponencial- Aceptación/Rechazo



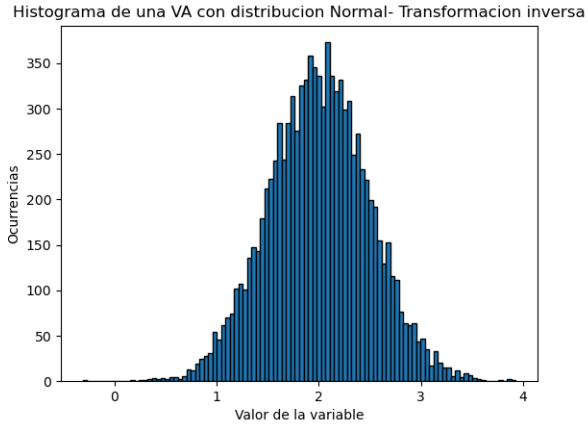
6.4. Distribución Exponencial- T. inversa



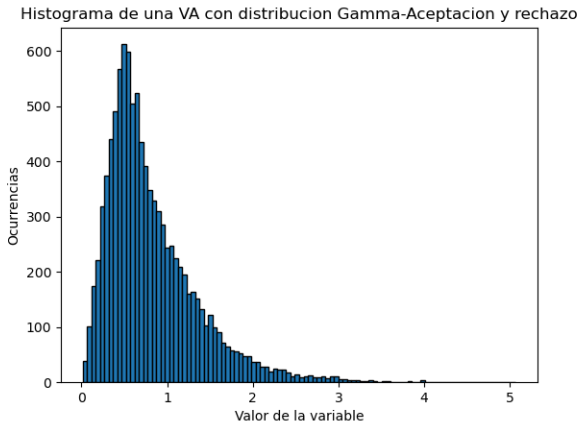
6.5. Distribución Normal- Aceptación/Rechazo



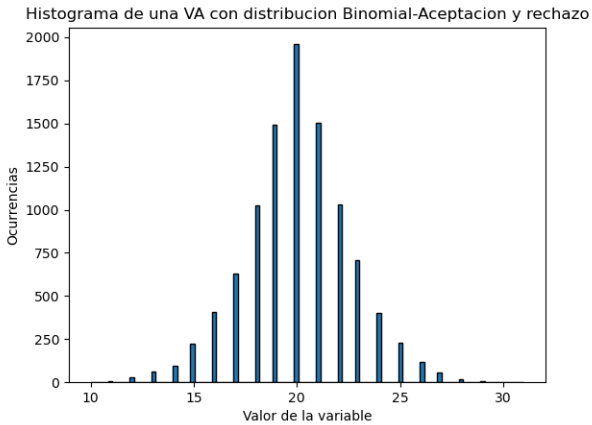
6.6. Distribución Normal- Transformada Inversa



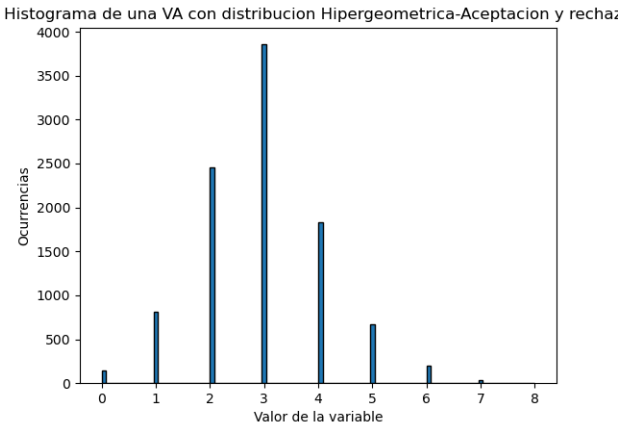
6.7. Distribución Gamma- Aceptación/Rechazo



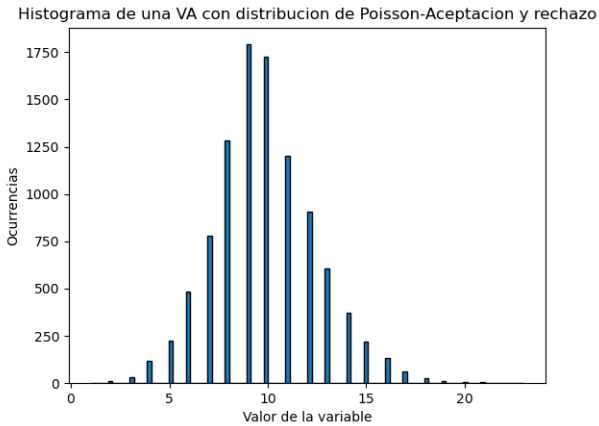
6.8. Distribución Binomial- Aceptación/Rechazo



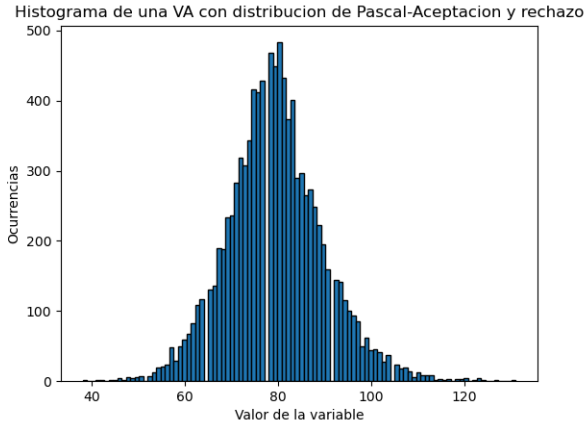
6.9. Distribución Hipergeometrica- A/R



6.10. Distribución Poisson- Aceptación/Rechazo



6.11. Distribución Pascal- Aceptación/Rechazo



6.12. Test Kolmogorov-Smirnov para distintas distribuciones[12][?]

El siguiente cuadro muestra los resultados de evaluar las distribuciones generadas con el Test Kolmogorov-Smirnov [14] usando intervalos de confianza del $95\,\%$ y del $\%97{,}5$.

Distribución	Test Kolmogorov-Smirnov(0.95)	Test Kolmogorov-Smirnov(0.975)
UniformeT	Pasa	Pasa
ExponencialT	Pasa	Pasa
NormalT	Pasa	Pasa
UniformeR	Pasa	Pasa
ExponencialR	Pasa	Pasa
NormalR	Pasa	Pasa
GammaR	Pasa	Pasa
BinomialR	Pasa	Pasa
PoissonR	Pasa	Pasa
PascalR	Pasa	Pasa
HipergeometricaR	No pasa	No pasa

7. Conclusión

La generación de distribuciones a partir de la función de densidad acumulada inversa(Método de transformación inversa) no solo requiere menos código, trabaja mucho mas rápido que el método de Aceptación y Rechazo. Esto se debe a que el método de transformación inversa genera números directamente con un numero pseudoaleatorio $r_i \in [0, 1]$ dado. Lamentablemente no es posible aplicar este método con todas las distribuciones. En cambio el método de aceptación y rechazo es mas lento ya que para cada par de elementos r_i debe decidir si uno de ellos se puede aceptar o no, pero por otro lado este método se puede aplicar a toda distribución. Se recomienda usar transformación inversa siempre que sea posible (especialmente para generar distribuciones exponenciales que es donde mayor es la diferencia de tiempo para la generación) pero el método de aceptación y rechazo también es útil y a veces necesario.

8. Referencias

[1] Técnicas de Simulación en computadoras-Thomas Naylor

[2] Formulas de calculo probabilistico:

Probabilidad y estadística para ingeniería y ciencias 9º edición- Pearson
https://www.pearson.com/en-us/subject-catalog/p/
probability--statistics-for-engineers--scientists/P200000007119/9780137273546

[3] Elberly College of Science - Uniform Distribution

https://online.stat.psu.edu/stat414/lesson/14/14.6

[4] Wolfram Hypergeometric Distribution

https://mathworld.wolfram.com/HypergeometricDistribution.html

[5] Wikipedia - Hypergeometric Distribution

https://en.wikipedia.org/wiki/Hypergeometric_distribution

[6] Wikipedia -Distribución de Erlang

https://es.wikipedia.org/wiki/Distribuci%C3%B3n_de_Erlang

[7] Wikipedia - Negative binomial distribution

https://en.wikipedia.org/wiki/Negative_binomial_distribution

[8] Simulación estadistica- Ruben Casal

https://rubenfcasal.github.io/simbook/AR.html

[9] Función error e inversa de la función error

https://es.wikipedia.org/wiki/Funci%C3%B3n_error#Funci%C3%B3n_inversa

[10] Inversa de la funcion error- scipy

https://docs.scipy.org/doc/scipy/reference/generated/scipy.special.erfinv.html

[11] Función de distribución empírica- Universidad Humboldt de Berlin

https://wikis.hu-berlin.de/mmint/Basics:_Empirical_Distribution_Function/es

[12] Comparación de distribuciones: test Kolmogorov–Smirnov. Joaquin Amat Rodrigo

https://www.cienciadedatos.net/documentos/51_comparacion_distribuciones_kolmogorov%E2%80%93smirnov

[13] Funciones estadísticas- SciPy

https://docs.scipy.org/doc/scipy/reference/stats.html

[14] Números Pseudoaleatorios

https://github.com/Faus14/Simulacion_2023/blob/main/Generadores.pdf

Anexos

Anexo I: La función error [9]
Se conoce como **función error** o **función error de Gauss** y se la nota como **erf** a la variable compleja definida como:

$$erf(z) = \frac{2}{\sqrt{\pi}} \int_{-\infty}^{z} e^{-t^2} dt \tag{.1}$$

Si el argumento de la función es real, devuelve un numero real