Contents

Clase 4: Técnicas de Muestreo de Distribuciones de Probabilidad (Parte 1) 1

Parámetros **3**

Generar la muestra **3**

Calcular la media y la varianza de la muestra **3**

Valores teóricos **3**

Imprimir los resultados **3**

Graficar un histograma de la muestra **3**

Parámetros **4**

Generar la muestra **4**

Graficar el histograma de la muestra y la función de densidad teórica **4**

“‘markdown

Clase 4: Técnicas de Muestreo de Distribuciones de Probabilidad (Parte 1)

Objetivos:

- Comprender la necesidad de técnicas de muestreo de distribuciones de probabilidad en simulación.
- Aprender y aplicar el método de la Transformada Inversa para generar variables aleatorias con distribuciones específicas.
- Aplicar el método de Aceptación-Rechazo para generar variables aleatorias con distribuciones más complejas.
- Identificar las ventajas y desventajas de cada método.

Contenido Teórico:

1. Introducción al Muestreo de Distribuciones de Probabilidad:

En la simulación, a menudo necesitamos generar valores aleatorios que sigan una distribución de probabilidad específica (no necesariamente uniforme). Estas distribuciones representan el comportamiento de diferentes variables en el sistema que estamos modelando. Por ejemplo, el tiempo entre llegadas de clientes a un banco podría seguir una distribución exponencial, o el número de piezas defectuosas en un lote podría seguir una distribución binomial. Por lo tanto, es crucial tener métodos para generar números aleatorios que sigan estas distribuciones.

2. Método de la Transformada Inversa:

Este método es aplicable cuando se conoce la función de distribución acumulativa (FDA) de la distribución deseada y se puede obtener su inversa analíticamente.

- **Principio:** El método se basa en la propiedad de que si U es una variable aleatoria uniformemente distribuida en el intervalo (0, 1), entonces $X = F^{-1}(U)$ es una variable aleatoria distribuida según la función de distribución $F(x)$.
- **Pasos:**
    1. Obtener la función de distribución acumulativa (FDA) $F(x)$ de la distribución deseada.
    2. Calcular la inversa de la FDA, $F^{-1}(y)$.
    3. Generar un número aleatorio u uniformemente distribuido en (0, 1).
    4. Calcular $x = F^{-1}(u)$. Este valor x es una muestra aleatoria de la distribución deseada.

- **Ejemplo: Distribución Exponencial:**

  La FDA de la distribución exponencial es $F(x) = 1 - e^{- x}$ para $x \quad 0$. Donde es el parámetro de tasa.

  1. Resolvemos para x en términos de $F(x)$ para obtener la inversa:
     - $F(x) = 1 - e^{- x} = u$
     - $e^{- x} = 1 - u$
     - $- x = ln(1 - u)$
     - $x = (-1/\ ) ln(1 - u)$
  2. Como U es uniforme en (0, 1), entonces (1 - U) también lo es. Por lo tanto, podemos simplificar la ecuación a:
     - $x = (-1/\ ) ln(U)$

  Por lo tanto, para generar una variable aleatoria exponencial, simplemente generamos un número aleatorio uniforme U y lo insertamos en la fórmula anterior.

- **Ventajas:** Es un método exacto y eficiente cuando la inversa de la FDA se puede calcular fácilmente.
- **Desventajas:** No es aplicable si la inversa de la FDA no tiene una forma analítica simple.

3. Método de Aceptación-Rechazo (Acceptance-Rejection Method):

Este método se utiliza cuando la FDA o su inversa son difíciles de calcular. Se basa en la idea de muestrear de una distribución "más simple" y luego aceptar o rechazar la muestra con una cierta probabilidad.

- **Requisitos:**
  - Conocimiento de la función de densidad de probabilidad (FDP) $f(x)$ de la distribución deseada.
  - Una función de densidad de probabilidad $g(x)$ "más simple" de la cual se pueda muestrear fácilmente (distribución propuesta o envolvente).
  - Una constante c tal que $f(x) \quad c\ g(x)$ para todo x. Esto significa que la función $c\ g(x)$ debe "envolver" a la función $f(x)$.
- **Pasos:**
  1. Generar una muestra y de la distribución $g(x)$.
  2. Generar un número aleatorio u uniformemente distribuido en (0, 1).
  3. Si $u \quad f(y) / (c * g(y))$, aceptar la muestra y como una muestra de la distribución $f(x)$. De lo contrario, rechazar la muestra y volver al paso 1.
- **Ejemplo: Generar valores de una distribución triangular:**

  Supongamos que deseamos generar números aleatorios de una distribución triangular con función de densidad: `f(x) = x para 0 <= x <= 1 f(x) = 2 - x para 1 < x <= 2 f(x) = 0 en otro caso`

  Podemos usar una distribución uniforme entre 0 y 2 como distribución envolvente g(x) = 1/2 para 0 <= x <= 2, y g(x) = 0 en otro caso. El valor de c sería 1, ya que el máximo de f(x) es 1 y el máximo de g(x) es 1/2, entonces 1 <= c * (1/2) implica que c = 2. Es decir, $f(x) <= 2 * g(x)$ para todo x.

  1. Generar `y ~ U(0, 2)`
  2. Generar `u ~ U(0, 1)`
  3. Si `u <= f(y) / (2 * (1/2)) = f(y)`, aceptar `y`, sino rechazar.
- **Ventajas:** Puede utilizarse para generar muestras de distribuciones complejas para las cuales otros métodos no son aplicables.
- **Desventajas:** Puede ser ineficiente si la distribución envolvente $g(x)$ no se ajusta bien a la distribución deseada $f(x)$, lo que resulta en una alta tasa de rechazo. La elección de $g(x)$ y c es crucial para la eficiencia.

Ejemplos/Casos de Estudio:

1. **Simulación de un Sistema de Inventario:** Un sistema de inventario donde la demanda diaria sigue una distribución de Poisson. Necesitamos generar números aleatorios que sigan la distribución de Poisson para simular la demanda diaria. Si (tasa promedio de demanda) es pequeña, podemos utilizar el método de la transformada inversa (derivando la FDA y luego su inversa). Si es grande, es más eficiente utilizar el método de Aceptación-Rechazo.
2. **Simulación de Tiempo de Reparación:** El tiempo de reparación de una máquina sigue una distribución Weibull. La distribución de Weibull tiene una FDA complicada, pero su inversa es derivable. Utilizar la transformada inversa para simular los tiempos de reparación.

Problemas Prácticos/Ejercicios con Soluciones:

1. **Generación de Variables Aleatorias Exponenciales:** Escribe un programa en Python (o cualquier otro lenguaje) que genere 1000 variables aleatorias que sigan una distribución exponencial con = 2 utilizando el método de la transformada inversa. Calcula la media y la varianza de la muestra generada y compárala con los valores teóricos (media = 1/ , varianza = 1/ ^{2}).

   “‘python import numpy as np import matplotlib.pyplot as plt

   def generar_exponencial_inversa(n, lam): """Genera n variables aleatorias exponenciales usando la transformada inversa."""" u = np.random.uniform(0, 1, n) x = (-1/lam) * np.log(u) return x

   # Parámetros

   n = 1000 lam = 2

   # Generar la muestra

   muestra = generar_exponencial_inversa(n, lam)

   # Calcular la media y la varianza de la muestra

   media_muestra = np.mean(muestra) varianza_muestra = np.var(muestra)

   # Valores teóricos

   media_teorica = 1/lam varianza_teorica = 1/(lam**2)

   # Imprimir los resultados

   print(f”Media de la muestra: {media_muestra}”) print(f”Varianza de la muestra: {varianza_muestra}”) print(f”Media teórica: {media_teorica}”) print(f”Varianza teórica: {varianza_teorica}”)

   # Graficar un histograma de la muestra

   plt.hist(muestra, bins=30, density=True, alpha=0.6, color=’g’) plt.title(’Histograma de Variables Aleatorias Exponenciales’) plt.xlabel(’Valor’) plt.ylabel(’Frecuencia’) plt.show() “‘
2. **Implementación de Aceptación-Rechazo:** Implementa el método de Aceptación-Rechazo en Python para generar números aleatorios de una distribución Beta(2, 5) utilizando una distribución

uniforme como distribución envolvente. Grafica el histograma de la muestra generada y compárala con la forma teórica de la distribución Beta(2, 5).

“‘python import numpy as np import matplotlib.pyplot as plt from scipy.stats import beta

def generar_beta_aceptacion_rechazo(n, alpha, beta_param): """Genera n variables aleatorias Beta(alpha, beta_param) usando Aceptación-Rechazo.""" samples = [] num_intentos = 0 while len(samples) < n: num_intentos += 1 # Usamos una uniforme U(0, 1) como g(x) y = np.random.uniform(0, 1) u = np.random.uniform(0, 1)

```
    # Calculamos f(y) y g(y)
    f_y = beta.pdf(y, alpha, beta_param) #función beta de scipy
    g_y = 1  # La función uniforme U(0, 1) tiene pdf = 1

    # Encontramos c tal que f(x) <= c * g(x) para todo x
    # En este caso, podemos usar el máximo de la función beta como c
    c = beta.pdf( (alpha - 1) / (alpha + beta_param - 2) , alpha, beta_param) # Máximo de la beta.

    # Condición de Aceptación-Rechazo
    if u <= f_y / (c * g_y):
        samples.append(y)

print(f"Número de intentos: {num_intentos}")
return np.array(samples)
```

Parámetros

n = 1000 alpha = 2 beta_param = 5

Generar la muestra

muestra = generar_beta_aceptacion_rechazo(n, alpha, beta_param)

Graficar el histograma de la muestra y la función de densidad teórica

x = np.linspace(0, 1, 100) plt.hist(muestra, bins=30, density=True, alpha=0.6, color=’b’, label=’Muestra generada’) plt.plot(x, beta.pdf(x, alpha, beta_param), ’r-’, label=’Distribución Beta Teórica’) plt.title(’Generación de Distribución Beta(2, 5) con Aceptación-Rechazo’) plt.xlabel(’Valor’) plt.ylabel(’Densidad’) plt.legend() plt.show() “‘

Materiales Complementarios Recomendados:

- **Libros:**
  - ”Simulation Modeling and Analysis” de Averill M. Law
  - ”Discrete-Event System Simulation” de Jerry Banks et al.
- **Artículos:** Buscar artículos académicos en bases de datos como IEEE Xplore o ACM Digital Library sobre técnicas de muestreo en simulación.
- **Recursos en línea:**
  - Vídeos de YouTube explicando los métodos de transformada inversa y aceptación-rechazo.
  - Tutoriales en línea sobre la implementación de estos métodos en diferentes lenguajes de programación.
- **Documentación de librerías:** Revisar la documentación de numpy y scipy en Python para funciones relacionadas con la generación de números aleatorios y distribuciones de probabilidad. “‘