Muestreo de variables aleatorias

Clase nro. 5

CURSO 2010

Muestreos

- Los problemas que tratamos son estocásticos.
- No podemos predecir la conducta de los elementos del sistema, pero sí podemos enumerar los resultados posibles de ellas.
- Entonces utilizamos técnicas de **muestreo de distribuciones.**

Muestreos

- Generación de **números aleatorios**.

  Se utilizan para obtener valores *independientes* de variables aleatorias.

- Muestreo de **variables aleatorias**.

  Se obtienen a partir de *números aleatorios*.

- Muestreo a partir de **histogramas**.

  Cuando los datos no se pueden ajustar a ninguna *distribución teórica paramétrica*, se utiliza la frecuencia relativa de las observaciones, representadas por histogramas (*distribuciones empíricas*).

Muestreos

- Para simular el comportamiento del sistema utilizamos valores de las variables aleatorias (muestreados, sorteados).

- Si queremos observar el comportamiento general del sistema alcanza con realizar una sola corrida.

- Si queremos obtener medidas o valores estadísticamente válidos deben realizarse varias corridas con distintos números aleatorios que generarán distintos valores de las distribuciones muestreadas.

Generación de números aleatorios

- Rol preponderante en el proceso de simulación.
- Para simular necesitamos de números aleatorios como semillas para generar muestras de v.a.
- Características de un generador de nros aleatorios:

  1) Muestrea valores de Distribución Uniforme.

  2) Asegura la NO Correlación Serial.

  3) Otras (Law y Kelton, 1992; Banks *et. al.*, 2001).

Propiedades de los Números Aleatorios

1) Distribución Uniforme.

Cualquier número que pertenezca al rango de interés debe tener la misma probabilidad de resultar sorteado.

2) NO Correlación Serial.

La aparición de un número en la secuencia, no afecta la probabilidad de sortear otro (o el mismo) número.

Ejemplo

La sucesión 1,2,3,4,5,1,2,3,4,5...

podríamos decir es uniforme

pero

está correlacionada.

Existen Tests que verifican las condiciones de uniformidad y correlación serial.

Procedimientos para generar números aleatorios

1. Utilización de tablas.
2. Dispositivos especiales.
3. Procedimientos, funciones que generan números pseudoaleatorios.

Tablas de números aleatorios

Se generan con métodos aleatorios puros mediante ruletas, extracción de números al azar, dados, etc.

La secuencia generada se carga en la memoria de la computadora. La compañía RAND (Research & Development) publicó una tabla de un millón de números en 1955.

Ventajas: son números aleatorios puros.

Desventajas:

- la sucesión de números es finita.
- hay que cargar la tabla en memoria.
- ocupa mucha memoria (actualmente no es un problema).

Dispositivos especiales

En base a algún circuito o mecanismo de la computadora (reloj p.ej) se generan números que son puramente aleatorios.

El método básicamente consiste en interrumpir un proceso uniforme aleatoriamente. Es esencialmente lo que ocurre cuando la bola cae en un casillero de la ruleta.

Ventajas: son números aleatorios puros.

Desventajas: si se desea generar la misma secuencia más de una vez, es necesario grabarla, no siempre podremos repetir la misma secuencia en caso de ser necesario.

Números pseudoaleatorios

Imitan los valores de una variable aleatoria uniforme. Cumplen los tests de ajustes como si fueran esa variable aleatoria.

Se generan a través de una fórmula.

Se usan como semilla para generar valores de variables aleatorias (discretas, continuas).

Pseudoaleatorios, porque se obtienen realizando un conjunto de operaciones a partir del número generado en algún paso anterior.

Ventaja: método muy veloz y barato.

Desventaja: son de período finito.

Números pseudoaleatorios

Tanto la **secuencias** como las **subsecuencias** de los números generados deben cumplir las hipótesis de:

1) Distribución **Uniforme.**

2) **Independencia** (no correlación serial).

Además:

4) deben ser secuencias **largas** y sin huecos **(densas)**

5) **algoritmos rápidos.**

Método Centros Cuadrados

Se elije un número, se lo eleva al cuadrado, luego se toman los dígitos del centro como el siguiente número; y se repite el procedimiento.

Ejemplo: **2061: 4247721**

2477: 6135529

1355: ...

Desventaja: secuencia por lo general corta.

Este ejemplo, genera 34 números pasando luego a sortear siempre 0. El 2500 genera siempre el 2500.

A veces, con grandes números se puede llegar a generar secuencias de 100.000 números diferentes.

Método Congruencial Lineal

Este es el método utilizado por excelencia.

Se basa en la siguiente recurrencia:

$$n_i = (a \cdot n_{i-1} + c) \bmod m = f(n_{i-1})$$

Si se quiere obtener números Uniformes en (0,1) se normaliza el resultado:

$$U_i = n_i / m$$

Método Congruencial Lineal

Ejemplos:

a) a = 3, c = 0, m = 5 y $n_0 = 4$; 2, 1, 3, 4, 2, 1 ....

b) a = 3, c = 0, m = 9 y $n_0 = 4$; 3, 0, 0, ...

En el MCL, si se repite un número ya se repite toda la secuencia.

Ventajas:

- **utiliza poca memoria y es muy rápido.**
- **fácil de volver a generar la misma secuencia, guardando un solo número, (alcanza con partir desde la misma semilla: n_0).**

Método Congruencial Lineal

Importante: la velocidad de generación, y por sobre todo el largo de la secuencia dependen de la elección de las constantes a, c, m y la semilla n_0.

Reglas que aseguran un **ciclo maximal** (Knuth):

1) *c* y *m* deben ser primos relativos (sin factores comunes)

2) si *p* es factor primo de *m*, entonces elegir $a = 1 \pmod{p}$

3) si 4 es factor de *m*, elegir $a = 1 \pmod 4$

Pascal-SIM genera según la precisión de la computadora:

16-bit: $n_{i+1} = f(n_i) = (3993 \cdot n_i + 1) \bmod 32\,767$

32-bit: $n_{i+1} = f(n_i) = (16\,807 \cdot n_i + 0) \bmod 2147\,483\,647$

Método Mersenne Twister (MT)

- **Los generadores en base al método congruencial lineal son muy utilizados, pero muchos de ellos tienen un período mucho más corto que el que uno desearía o necesita.**
- **En 1997, Matsumoto presentó los generadores MT Son de período "largo" y "rápidos".**
- **Hay varias implementaciones y es utilizado para SED y Monte Carlo.**
- **Este método es el usado por EoSimulator.**

Método Mersenne Twister

Utiliza N celdas para generar los números aleatorios, bajo la siguiente recurrencia:

$$\mathbf{X_{k+n} = X_{k+m} \oplus (X^u_k | X^l_{k+1})A, \qquad (k = 0, 1, \ldots) \quad (I)}$$

Donde:

X: entero de w bits.

$\oplus$: XOR.

|: Concatenación de cadenas de bits.

X^j: fragmento de j bits de X. u son los bits más significativos y l los menos significativos.

A: Matriz $w \times w$.

n: grado de recurrencia con $1 \leq m \leq n$

Método Mersenne Twister

El método trabaja a más bajo nivel y por eso al utilizar operaciones lógicas en cadenas de bits (XOR, AND, OR), entonces son generadores rápidos.

Mt19937 es un ejemplo, tiene un periodo de 2^{19937}-1.

El método ha sido validado y ha pasado tests exigentes (Die Hard; Marsaglia, 1985).

Ref.: http://en.wikipedia.org/wiki/Mersenne_twister#References

Homepage http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/emt.html

Streams (torrentes)

Un generador de números aleatorios que comience con la misma semilla, siempre producirá el mismo torrente o secuencia de números.

Diferentes semillas generarán diferentes secuencias. Si las semillas se eligen con valores no cercanos (en el ciclo del generador), entonces las secuencias de números generados (torrentes) parecerán y actuarán como números aleatorios independientes entre sí con lo que colaborarán en la generación de v.a. independientes entre sí.

Streams (torrentes)

Al comparar los efectos de distintas políticas en un modelo, (p. ej el número de camas) es importante que las corridas del modelo, se ejecuten con los mismos valores (tiempos) en las actividades y las variables de decisión (v.a.).

Cuando generamos muestras de una v.a., si utilizamos la misma secuencia de números pseudo-aleatorios, generamos la misma secuencia de valores (muestras) de esas v.a.

Para obtener valores esperados de las v.a., se realizan varias corridas. En cada corrida deben ser usados diferentes torrentes de números e independientes entre sí (n_0 distinto para c/v.a.). El valor de la muestra debe ser dado en un intervalo de confianza.

Streams (torrentes)

Pascal Sim

make-stream: inicializa 32 streams.

original_seed: vector con las semillas iniciales.

seeds: vector con el valor actual de las semillas de cada stream.

Un stream puede ser reseteado individualmente:

seeds(j) := original_seeds(j).

function rnd *(s : stream_num) : real;*

genera el siguiente número de la secuencia, actualiza seed.

Streams (torrentes)

EOSimulator

Las semillas se setean manualmente,

de forma centralizada

(*Experiment::setSeed*)

o individualmente

(*Distribution::setSeed*).

Cada instancia de distribución tiene su propio generador de números pseudoaleatorios (que genera su propio torrente).

Tests: Uniformidad e Independencia

Test de χ^2

Utilizado para probar la uniformidad de la secuencia de los números pseudoaleatorios (válido para ajustar otras funciones de distribución).

El método consiste en tomar *n* observaciones independientes de la variable aleatoria (en nuestro caso los números generados), que llamaremos

$$X_1, X_2, \ldots, X_n$$

Test de χ^2

El intervalo en el que varía la variable aleatoria se divide en K categorías,

Se conoce la probabilidad (teórica) P_s de que la v.a. muestree en cada categoría s.

Sea Y_s la cantidad de valores de X_i pertenecientes a la categoría s. Entonces:

$$Y_1 + Y_2 + ... + Y_k = n$$

$$P_1 + P_2 + ... + P_k = 1$$

Test de χ^2

Construimos el estimador $$V = \sum_{s=1}^{k} \frac{(Y_s - nP_s)^2}{nP_s}$$

Se demuestra que V es una v.a. con distribución χ^2 con k-1 grados de libertad para n (n debe ser grande para que el test sea válido).

Además, se debe cumplir que $nP_s > 5$ (por lo menos 5 observaciones en cada categoría).

Test de χ^2

Se calcula *V* y se analiza el mismo utilizando la tabla de χ^2 (Apéndice 5, Hillier y Lieberman) mediante el test de significación siguiente:

- Si $V > \chi^2(f)$ con f = k-1(valor de la tabla), rechazar la hipótesis.
- Si no, no rechazar la hipótesis.

Dado un nivel de significación (p.ej. de 0,05 equivalente al 95%) nos fijamos en la tabla el valor correspondiente a ese nivel y para *k*-1 grados de libertad.

Si *V* > el valor crítico de la tabla se rechaza la hipótesis y si no, no se rechaza.

Ejemplo

Tirando 96 veces un dado se obtuvieron cantidades de 1s, 2s, etc: 15, 7, 9, 20, 26, 19.

Se desea saber si el dado es simétrico,

la hipótesis es que $P_1, P_2, P_3 \ldots, P_6 = 1/6$,

n = 96, Y_1 =15, etc. n P_i = 16 suficientemente grande.

f = k-1 = 5. nivel de significación = 0.01 ; $\chi^2(5) = 15.1$

$V = (15-16)^2/16 + (7-16)^2/16 + (9-16)^2/16 +$

$(20-16)^2/16 + (26-6)^2/16 + (19-16)^2/16 = 16 > 15.1$

Por lo tanto se rechaza la hipótesis. (se supone que el dado no es simétrico)

Test serial

Sirve para probar la correlación serial de la secuencia de números observados.

Se agrupa la muestra en pares de valores los cuales estarán distribuídos uniformemente e independientes entre sí, en caso de verificar el test.

La muestra será de $2n$ valores.

Se consideran las n parejas X_{2j}, X_{2j+1} con $0 < j < n$

(X_0,X_1) (X_2,X_3) ..... (X_{2n-2},X_{2n-1})

1 2 n-1

Test serial

Cada pareja tiene probabilidad

$$P(X_{2j}, X_{2j+1}) = 1/K^2,$$

y tendremos K^2 categorías.

(que es la cantidad de combinaciones posibles de parejas de números uniformes).

Se aplica el test de χ^2 con K^2-1 grados de libertad con el mismo criterio que definimos anteriormente.

Sorteos de variables aleatorias

- Variables aleatorias discretas.

  General, Poisson.

- Variables aleatorias continuas.

  Exponencial negativa, Normal, log Normal.

Generación de v.a. discretas

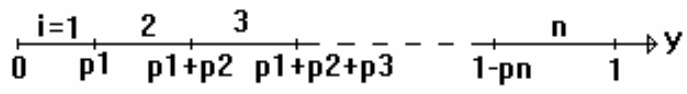
X es un v.a. que puede tomar los valores $x_1, x_2, \ldots, x_n$,

con probabilidad respectiva $p_1, p_2, \ldots, p_n$.

Dividimos el intervalo [0,1] en intervalos de longitud

$p_1, p_2, \ldots\ldots p_n$, donde se cumple

$$\sum_{i=1}^{n} p_i = 1$$



Generación de v.a. discretas

Al intervalo i le corresponden aquellos valores x_i que cumplen:

$$\sum_{j=0}^{i-1} p_j \leq x_i \leq \sum_{j=0}^{i} p_j$$

Sorteo de una muestra de X

1. Se toma un valor $u = U(0,1)$.
2. Tomamos el punto $y = u$.
3. Si este punto aparece en el intervalo correspondiente al número i aceptamos que $X= x_i$ en este sorteo.

Generación de v.a. discretas

Validez **del método**:

la probabilidad de que u pertenezca a uno de los intervalos es igual a la longitud del mismo, ya que u es U(0,1).

$$P(0 < u < p_1) = p_1$$

$$P(p_1 < u < p_1+p_2) = p_2$$

.

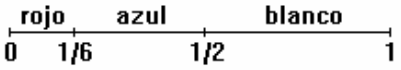
.

$$P(1-p_n < u < 1) = p_n$$

Ejemplo: v.a. discreta

Una v.a. X puede tomar los valores: rojo con prob. 1/6, azul con prob. 1/3 y blanco con prob. 1/2.

Construyo el esquema:



Sorteo u =	0,1212	X = rojo
	0,9432	blanco
	0,6111	blanco
	0,4343	azul

Distribución Poisson

Cantidad de arribos en un intervalo de tiempo.

Función de probabilidad $f(x) = \lambda^x \exp(-\lambda)/x!$

Función de distribución cumple $F(x+1) = F(x) + f(x+1)$, propiedad que se utiliza para la obtención de muestras (Ver seudocódigo en Davies y O'Keefe)

Generación de v.a. continuas

Queremos generar valores de X v.a con distribución:

$$F_X(x) = \int_{-\infty}^{x} f_X(t)dt$$

Método de la transformación inversa.

Se iguala la función de distribución de X, a una v.a. U uniforme (0,1).

$U = F_x(X)$ $\qquad$ $X = F_x^{-1}(U)$

El problema se reduce a encontrar una expresión analítica de la función inversa, de modo de despejar X en función de U.

Generación de v. a. continuas

$U = F_x(X)$ $\qquad$ $X = F_x^{-1}(U)$

Generamos una v.a. U de distribución U(0,1), según los métodos de sorteo de números aleatorios (pseudoaleatorios).

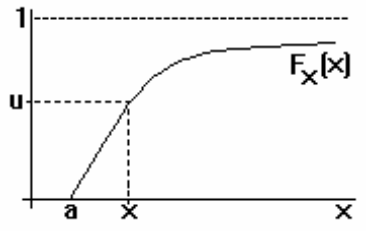
Encontramos una expresión analítica de la función inversa; la resolvemos en función de los valores sorteados de U.

Generación de v. a. continuas

Si F(x) es continua y estrictamente creciente,

$0 < F(x) < 1$, entonces la función inversa siempre existe.

Interpretación Geométrica:



Método de la Transformación Inversa

Problema:

aún cuando exista una expresión analítica de la inversa de la distribución, su cálculo puede insumir cantidades importantes de recursos y tiempo.

Distribución Uniforme

v.a. Uniforme U(a,b) $\quad f(x) = 1/(b - a)$

$$F_X(x) = \int_a^x \frac{1}{b-a}\,dt \qquad \text{y además} \qquad u = \int_a^x \frac{1}{b-a}\,dt$$

$u = (x - a)/(b - a)$, $x = a + u(b - a)$

x es U(a,b) con u v.a. U(0,1) (número pseudoaleatorio)

Distribución Exponencial

Tiempos entre dos sucesos Poisson.

$F_x(x) = 1 - e^{-\lambda x} \Rightarrow u = 1 - e^{-\lambda x} \Rightarrow e^{-\lambda x} = 1 - u \Rightarrow$

$-\lambda x = \ln(1 - u) \Rightarrow x = - 1/\lambda \ln(u)$

$(1 - u)$ es equivalente a u (u es U(0,1)).

Entonces $x = - 1/\lambda \ln(u)$ es una muestra de una v.a. exponencial de parámetro λ, a partir de un muestra u v.a. U (0,1) (número pseudoaleatorio).

Distribución Normal

Actividades que varían estocásticamente alrededor de un valor medio.

Ejemplo: tiempo que lleva arreglar una máquina con una falla bien conocida.

Observar que esta distribución no debiera ser usada para valores de tiempos, ya que no tiene cota inferior, puede tomar valores negativos (los tiempos son positivos y tienen valor mínimo 0 (cero)). Si se utiliza, deben eliminarse valores negativos.

No existe expresión para la inversa, método de Box y Muller (ver Davies y O'Keefe).

Distribución log Normal

Usada para describir **tiempos en filas de espera.**

Tiene dos parámetros media ***m*** y desviación est. ***s***.

Una muestra es log Normal, si su logaritmo corresponde a una muestra N(μ,σ), sus parámetros se relacionan de la siguiente manera:

$$\mu = \log_e m - 1/2 \log_e [(s/m)^2 + 1]$$
$$\sigma^2 = \log_e [(s/m)^2 + 1)$$

Nota: error en el libro Davies y O'Keefe.

Pascal Sim y EOSimulator

Pascal Sim:

function **poisson** (m: real; s : **stream_num**) : cardinal;

function **rnd** (s : **stream_num**) : real;

function **uniform** (l, h : real; s : **stream_num**) : real;

function **negexp** (m: real; s : **stream_num**) : real;

function **normal** (m, sd : real; s : **stream_num**) : real;

function **log_normal** (m, sd : real; s : **stream_num**) : real;

Pascal Sim y EOSimulator

EOSimulator:

- Clase abstracta *Distribution* con operaciones *sample* (abstracta) y *setSeed*. Constructor recibe el generador de números pseudoaleatorios a utilizarse (representado por un label).
- Una clase concreta por cada distribución, define métodos para *sample*.
- Distribuciones disponibles:
  - LogNormalDist
  - NegexpDist
  - NormalDist
  - PoissonDist
  - UniformDist

Muestreo de histogramas

Si los datos no se ajustan a ninguna distribución conocida, las muestras deben tomarse de una distribución de probabilidades derivada de un histograma de frecuencias de actividades o tiempos de arribos.

Ver libro página 75.

Muestreo de histogramas

Como una v.a discreta, utilizando las frecuencias obtenidas (los porcentajes, fig 4.3).

Muestrear de la función acumulativa escalonada (equivalente anterior) (fig 4.4).

Muestrear en rango de tiempo continuo, mediante distribución acumulada continua (fig 4.5).

Si u está entre $F(x)$ y F(x+1) $\Rightarrow$ s está entre x y x+1

la distancia de s a x está en proporción a la distancia de u a $F(x)$ entonces $s = x + (u\text{-}F(x))/(F(x+1) - F(x))$

Resumen

Los tiempos de actividades y los distintos criterios de decisión son simulados mediante muestreo de distribuciones.

Los generadores de números pseudaleatorios proveen Streams de números en el rango de (0,1) Uniformes independientes y no correlacionados.

El método de transformación inversa se utiliza para para muestrear valores de distribuciones o de histogramas de frecuencias.

Otros métodos: Normal (Box-Muller), Log Normal.