Números Aleatorios

Generación de Números y Variables Pseudo-Aleatorias

Leslie Murray

Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Universidad Nacional de Rosario
Rosario, Argentina

Mayo, 2024

Números Aleatorios

- En la simulación estocástica de tipo Monte Carlo, se reproduce artificialmente la evolución de sistemas, a veces complejos.
- No siempre se conocen reglas estrictas ni funciones determinísticas que describan el comportamiento de los sistemas, muchas veces se cuenta sólo con indicios, aproximaciones o descripciones cualitativas.
- Reproducir artificialmente la evolución de un sistema supone, entre otras cosas, disponer de sucesiones de números aleatorios con distribuciones tales que sus valores coincidan (al menos aproximadamente) con los que se leerían al observar, en plena operación, las variables del sistema cuya evolución se busca reproducir.
- Los sistemas de cómputo (autómatas de estado finito) no son capaces de generar números aleatorios$^{(*)}$ sino números pseudo–aleatorios.

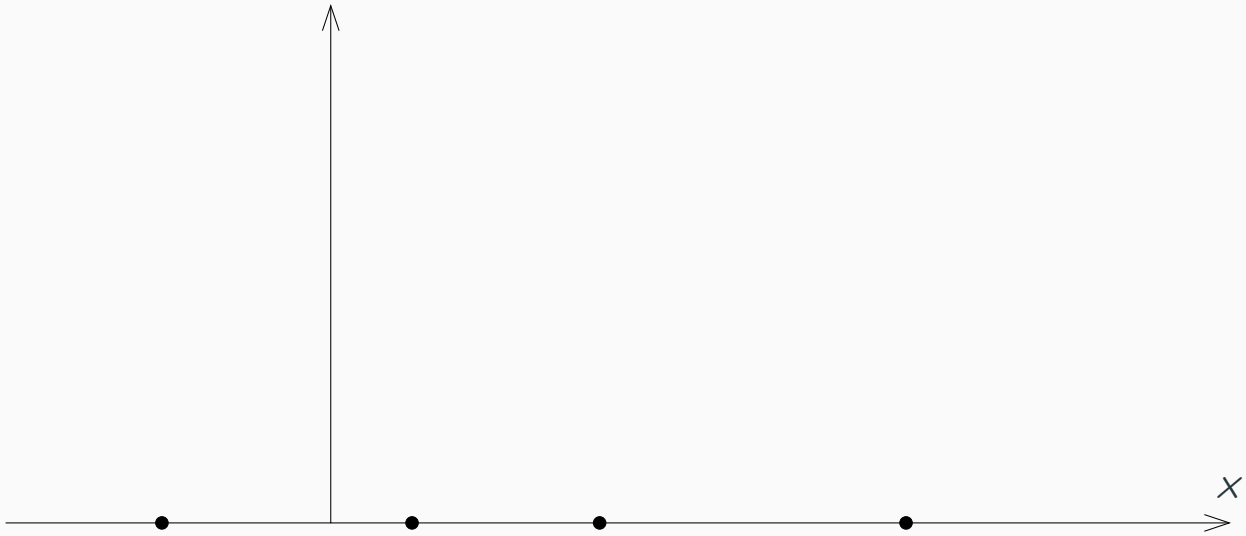
$^{(*)}$ No son capaces de generar números aleatorios como resultado de un proceso de cómputo.

Números Pseudo–Aleatorios

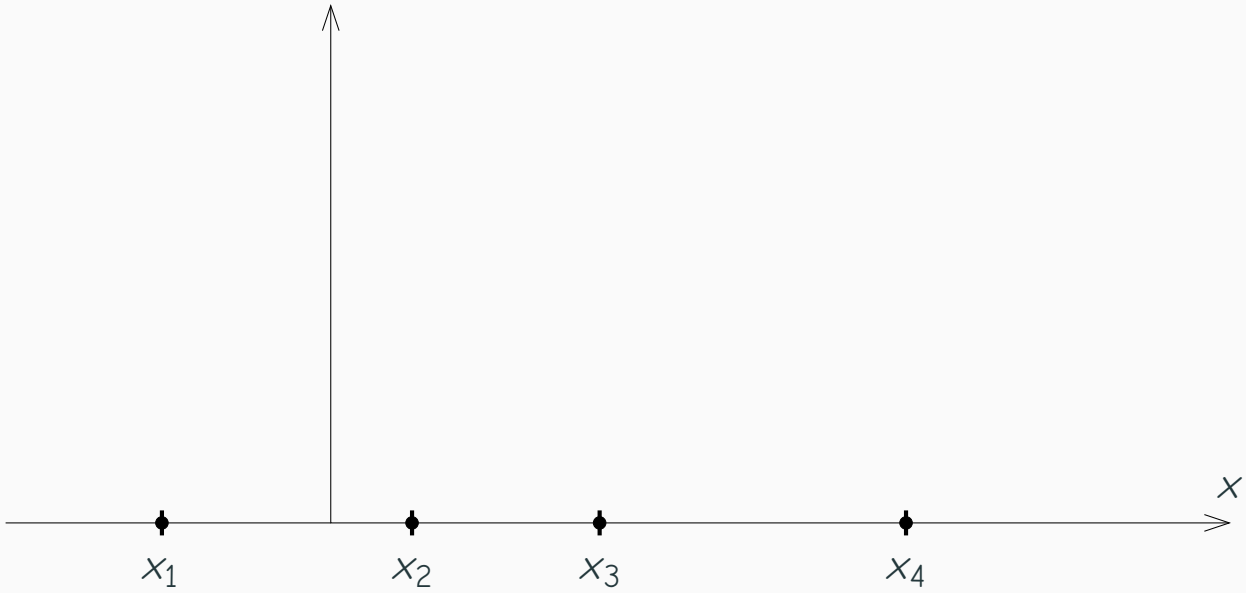
- Las secuencias de números pseudo–aleatorios son, en realidad, secuencias determinísticas (periódicas) generadas mediante un algoritmo.
- Las secuencias de números pseudo–aleatorios tienen algunas propiedades que las hacen "semejantes" a secuencias de números aleatorios:
  - No siguen un patrón o regla de generación evidente.
  - La correlación entre sus elementos es tan baja que dan la impresión de ser independientes
- La generación de números pseudo–aleatorios no requiere de elementos mecánicos ni de sistemas físicos y es, por lo tanto, más simple (eventualmente más rápida) que la generación de números aleatorios.
- **Una secuencia de números pseudo–aleatorios se puede repetir lo cual, en la experimentación mediante simulación, es extremadamente útil.**
- Los números pseudo–aleatorios pueden ajustarse a las mismas distribuciones que los números aleatorios.

La base para generar números pseudo-aleatorios con "cualquier distribución" es una secuencia de números pseudo-aleatorios uniformemente distribuidos en $[0,1]$.

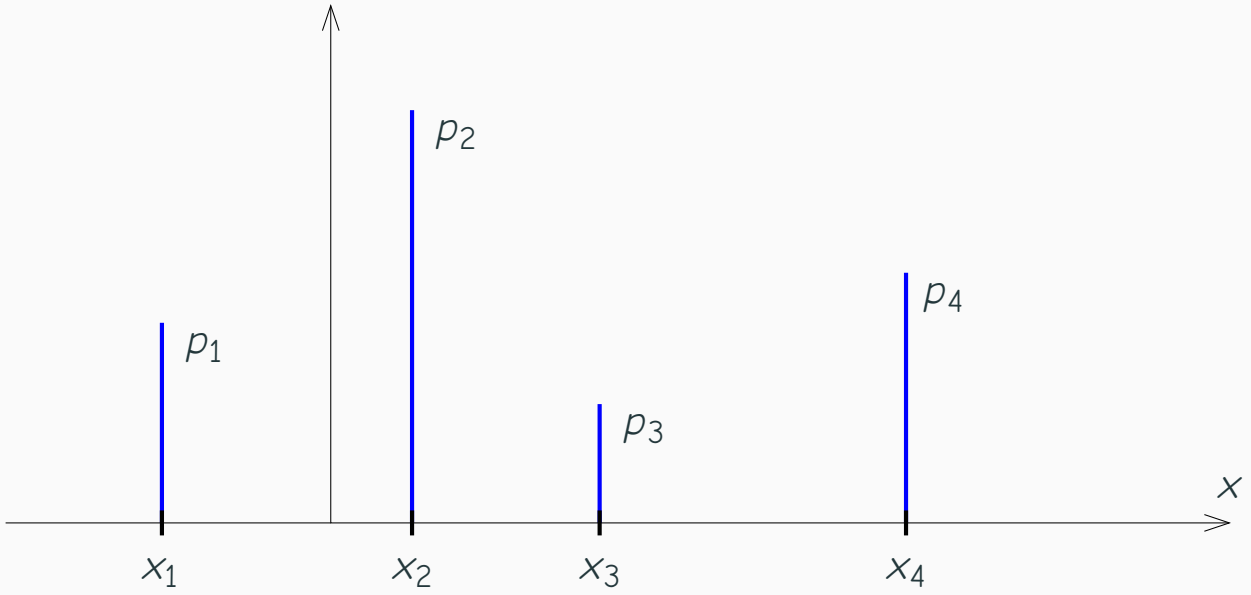
Distribución Discreta

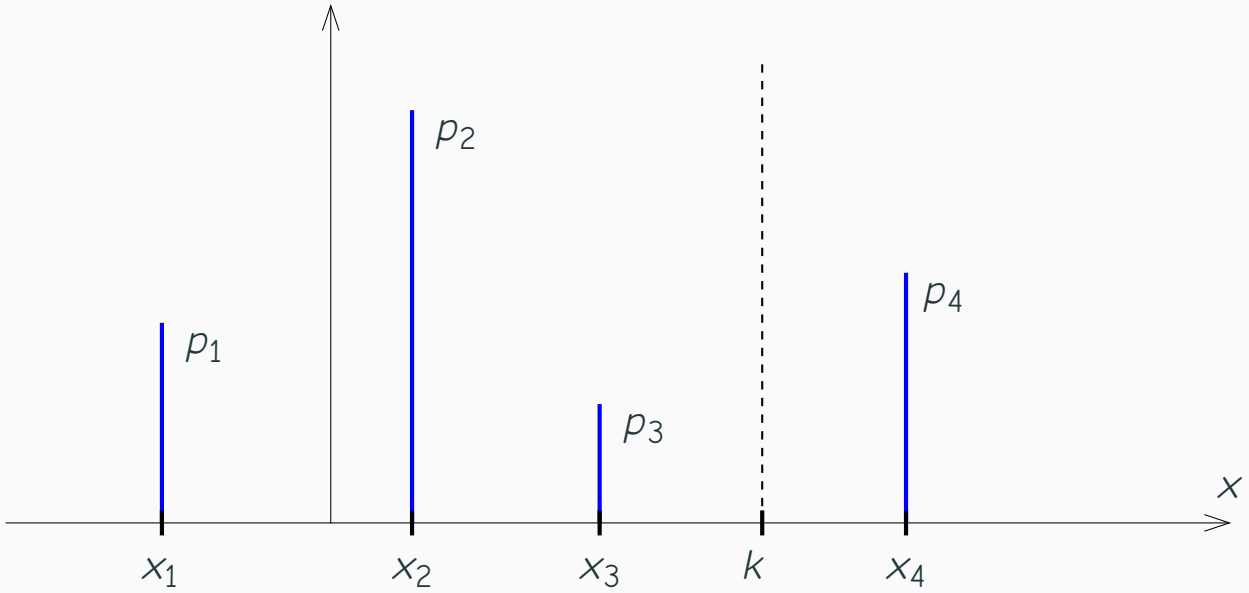


Distribución Discreta



Distribución Discreta

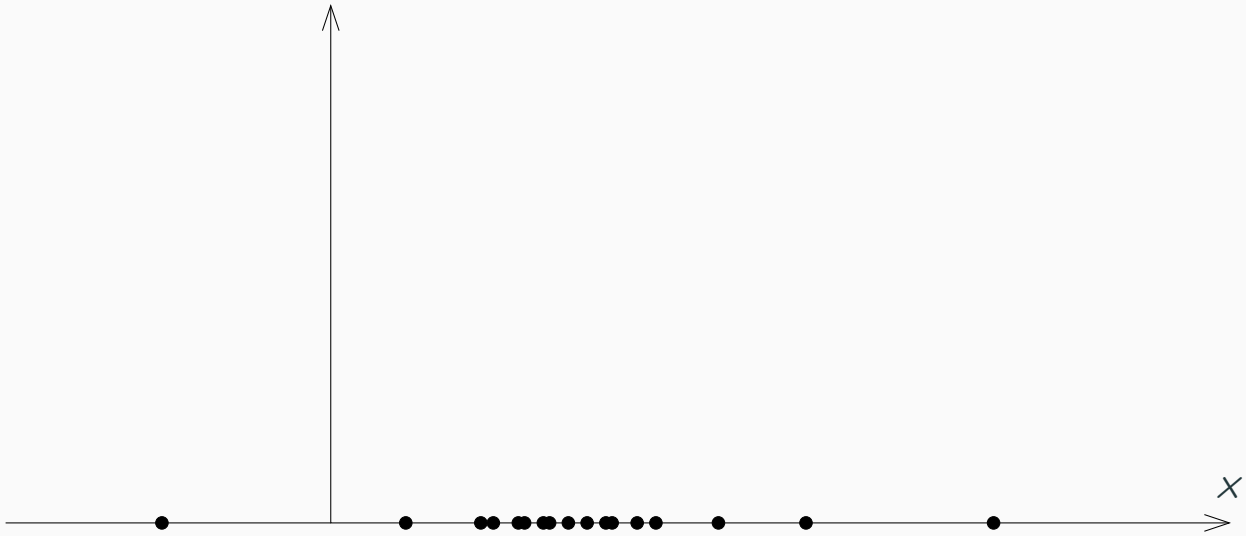




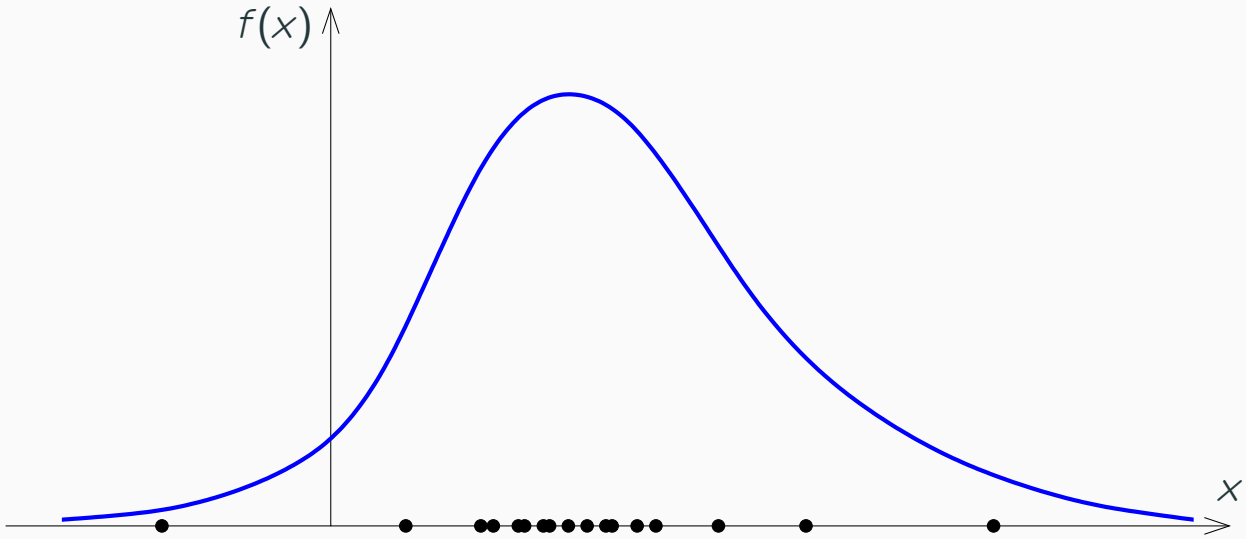
$$\sum_{i=1}^{4} p_i = 1$$

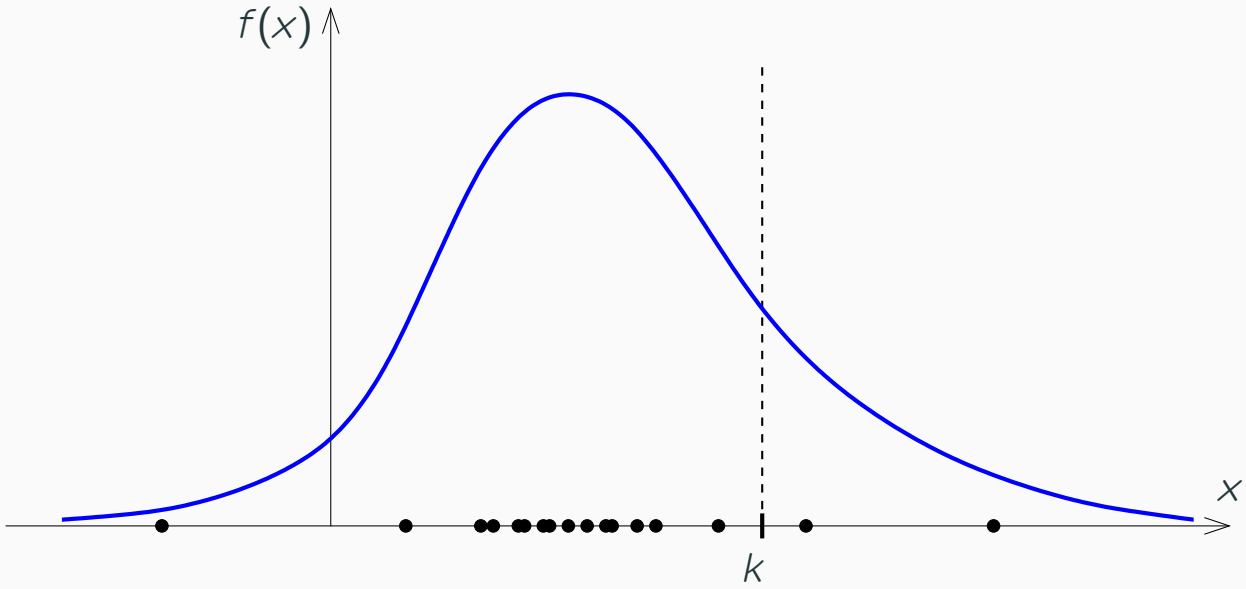
$$\sum_{i=1}^{3} p_i = \mathbb{P}\{X < k\}$$

Distribución Continua



Distribución Continua

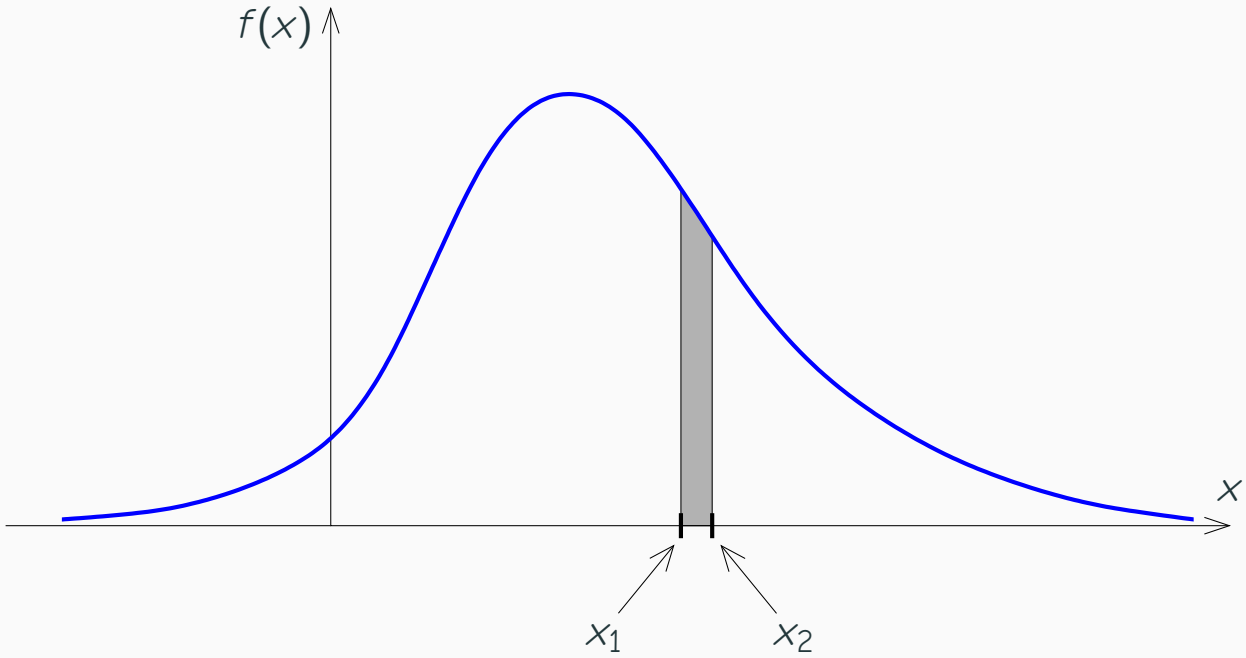




$$\int_{-\infty}^{+\infty} f(x)\,dx = 1$$

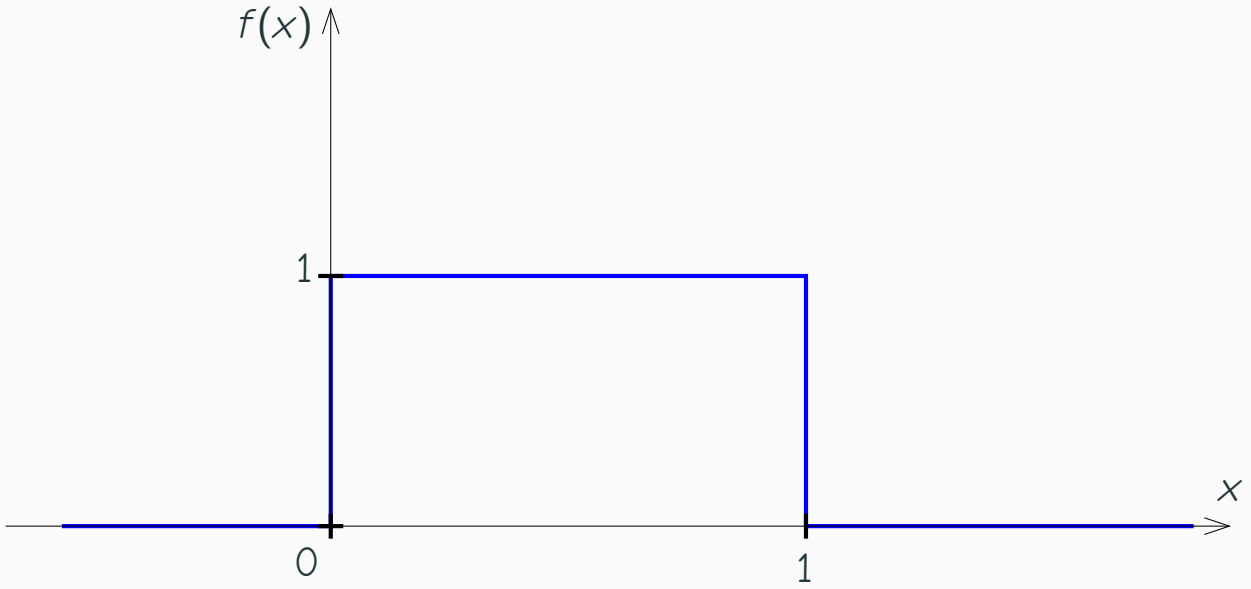
$$\int_{-\infty}^{k} f(x)\,dx = \mathbb{P}\{X < k\} = F(k)$$

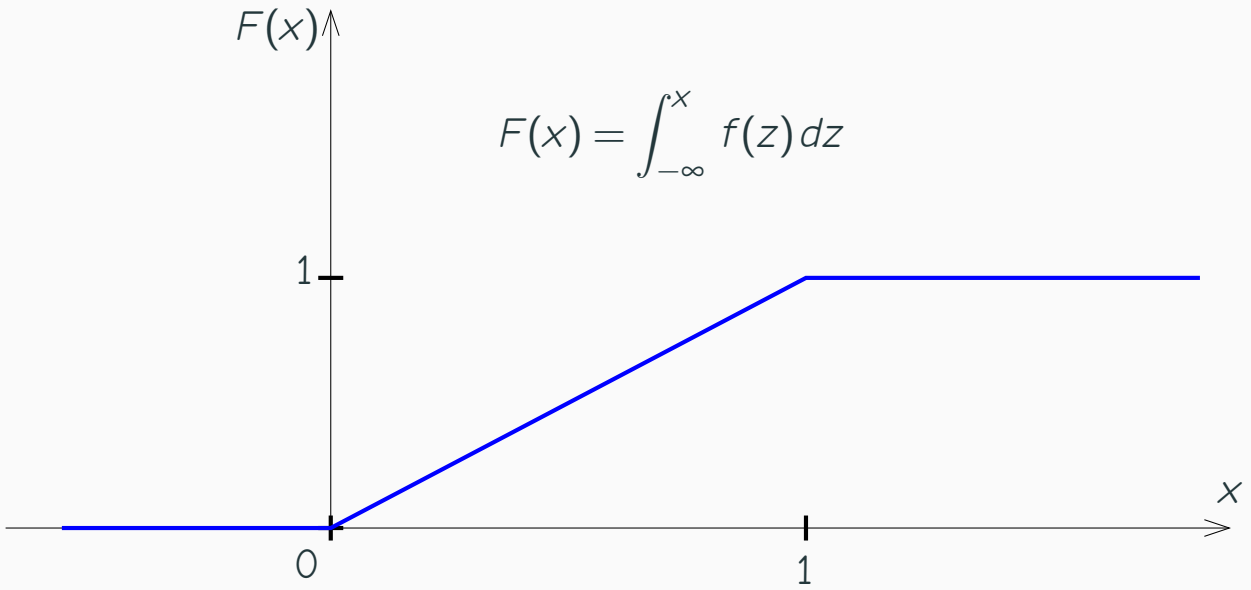
Distribución Continua



$$\mathbb{P}\{x_1 < X < x_2\} = \int_{x_1}^{x_2} f(x)\,dx$$

Distribución Continua Uniforme (0,1)





$$F(x) = \int_{-\infty}^{x} f(z)\,dz$$

Generador congruencial lineal

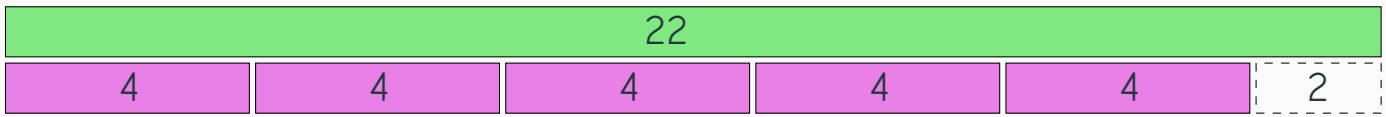
Uno de los algoritmos más conocidos y utilizados para generar una secuencia X_n, $n = 0,1,\ldots$ de números pseudo–aleatorios es el *generador congruencial lineal*:

$$X_{n+1} = (aX_n + c) \bmod m, \qquad n \geq 0$$

- a, c y m son constantes enteras.
- La recurrencia se inicia con un X_0 entero por lo que los X_n, $n > 0$, son enteros.
- La sucesión X_n toma valores en $\{0,1,\ldots,m-1\}$.
- La sucesión X_n es periódica de período, máximo, $m-1$.
- Los valores X_n son pseudo–aleatorios en $[0,m)$.
- Los valores X_n/m son pseudo–aleatorios en $[0,1)$.

La operación $x \bmod y$ devuelve el resto de la división x/y en números enteros.
Ejemplo:

División en números reales	$\rightarrow$	$22/4 = 5,5$
División en números enteros	$\rightarrow$	$22/4 = 5$
Módulo	$\rightarrow$	$22 \bmod 4 = 2$



EJEMPLO 1: $a = 3$, $c = 3$, $m = 5$ y $X_0 = 0$

$$X_{n+1} = (3X_n + 3) \bmod 5$$

$$X_0 = 0, \quad X_1 = 3, \quad X_2 = 2, \quad X_3 = 4, \quad X_4 = 0, \ldots$$

$$X_0/m = 0.0, \quad X_1/m = 0.6, \quad X_2/m = 0.4, \quad X_3/m = 0.8, \quad X_4/m = 0.0, \ldots$$

□

- Los valores de la sucesión pertenecen a $\{0,1,2,3,4\}$
- La sucesión es periódica, de período $m-1=4$.
- El valor inicial, X_0, se llama *semilla* (la misma *semilla* genera la misma secuencia).
- Evidentemente conviene elegir m muy grande (un valor típico para un computador de 32 bits es $2^{31}-1$).
- Son pocas las combinaciones $\{a,c,m\}$ que dan resultados buenos o muy buenos (muchas dan resultados muy malos).
- En el lenguaje C, *drand*48(), *lrand*48() y *mrand*48(), son generadores de este tipo, mientras que *srand*48(), *seed*48() y *lcong*48(), sirven para fijar la *semilla*.
- Hay muchos algoritmos específicos (sofisticados) que generan secuencias con distribución Unif(0,1) con mejores propiedades estadísticas que el congruencial lineal (por ejemplo: PCG, Mersenne–Twister, Xoroshiro128+).

Generación de una variable aleatoria X dada su distribución

Las sucesiones con distribución Unif(0,1) sirven para generar valores ajustados a:

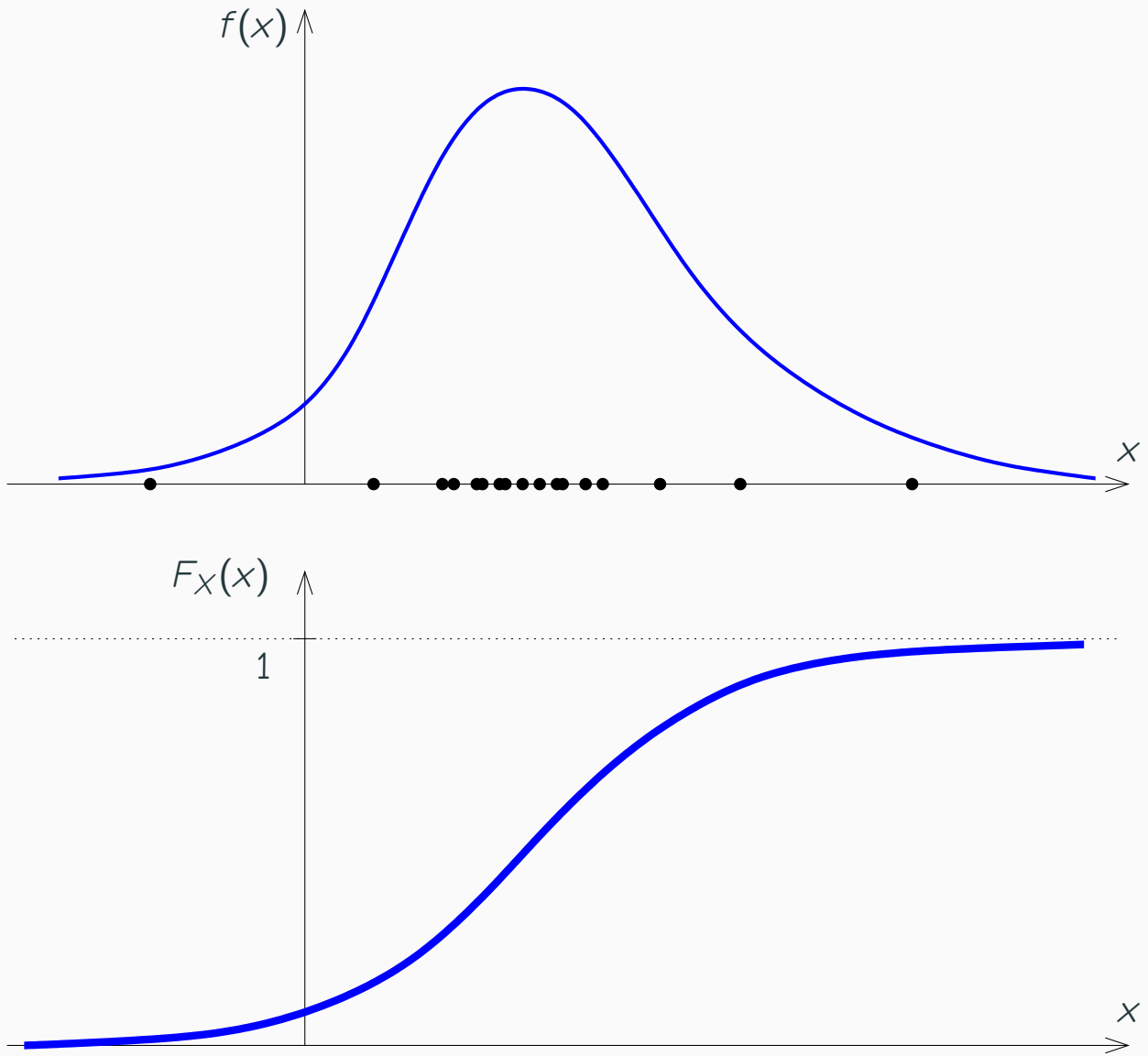
- una determinada *fdp* $f_X(x)$, en el caso de las v.a. continuas,
- una determinada función de masa de probabilidad, en el caso de las v.a. discretas.

En las secciones siguientes se presentan los métodos:

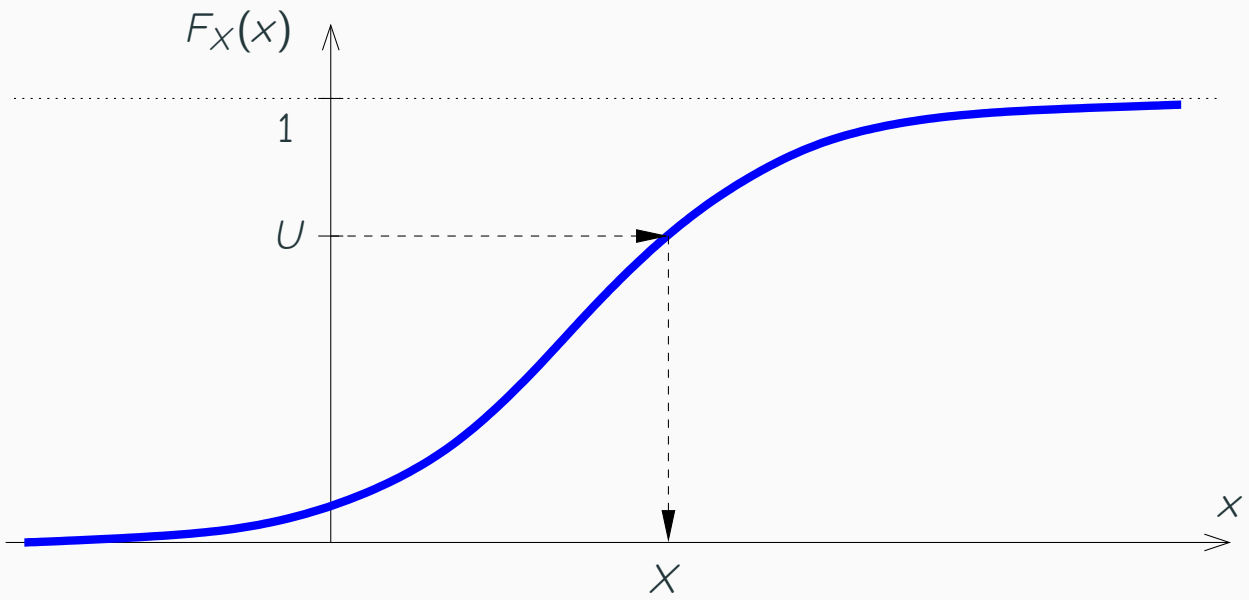
- Método de la Transformada Inversa – X continua,
- Método de la Transformada Inversa – X discreta,
- Método de Aceptación/Rechazo – X continua,

Nota: De aquí en más llamaremos *aleatorios* a números que son, en realidad, pseudo–aleatorios.

Generación de una variable aleatoria X dada su distribución



Método de la Transformada Inversa — X continua (I)



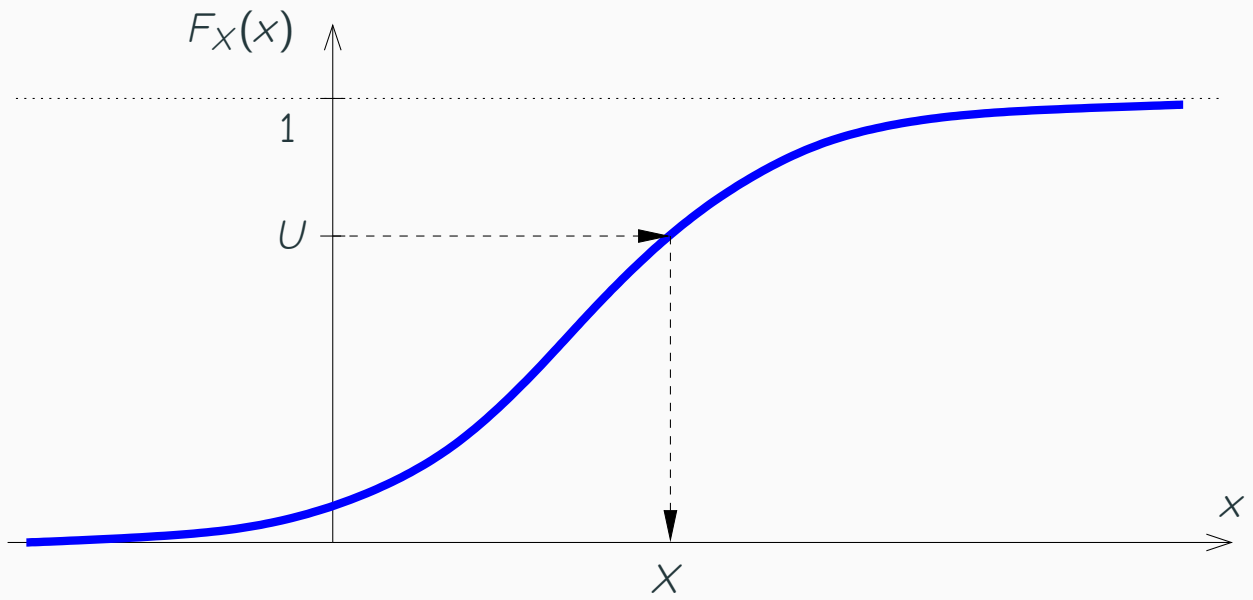
Si $X = F_X^{-1}(U)$, $U \sim \text{Unif}(0,1)$ entonces $X \sim f_X(x)$

Donde:

$$F_X^{-1}(y) = \inf\{x \ : \ F_X(x) \geq y\}, \qquad 0 \leq y \leq 1$$

El método es útil y eficiente sólo si es simple la determinación de la inversa de $F_X(x)$.

Método de la Transformada Inversa — X continua (II)

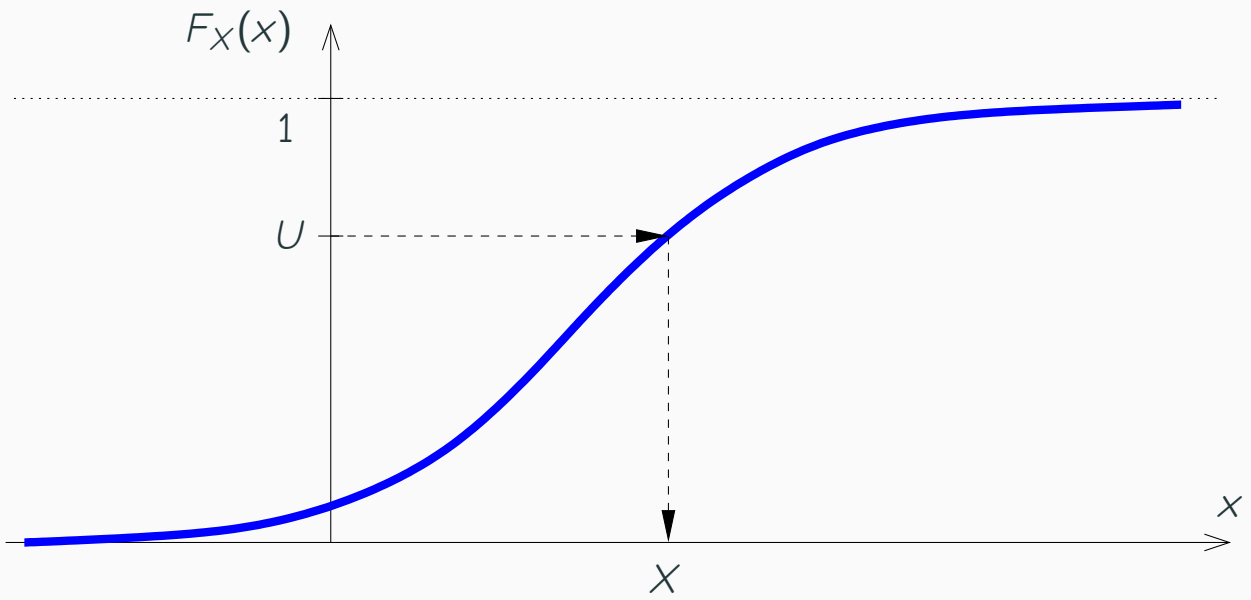


Si $X = F_X^{-1}(U)$, $U \sim \text{Unif}(0,1)$ entonces $X \sim f_X(x)$

Demostración:

$$\begin{aligned} F_X(x) &= \mathbb{P}\{X \le x\} \\ &= \mathbb{P}\{F_X^{-1}(U) \le x\} \quad \leftarrow \quad F_X(x) \text{ es monótona, } F_X^{-1}(U) \le x \Leftrightarrow U \le F_X(x) \\ &= \mathbb{P}\{U \le F_X(x)\} \\ &= F_X(x) \end{aligned}$$

Método de la Transformada Inversa — X continua (III)



Si $X = F_X^{-1}(U)$, $U \sim \text{Unif}(0,1)$ entonces $X \sim f_X(x)$

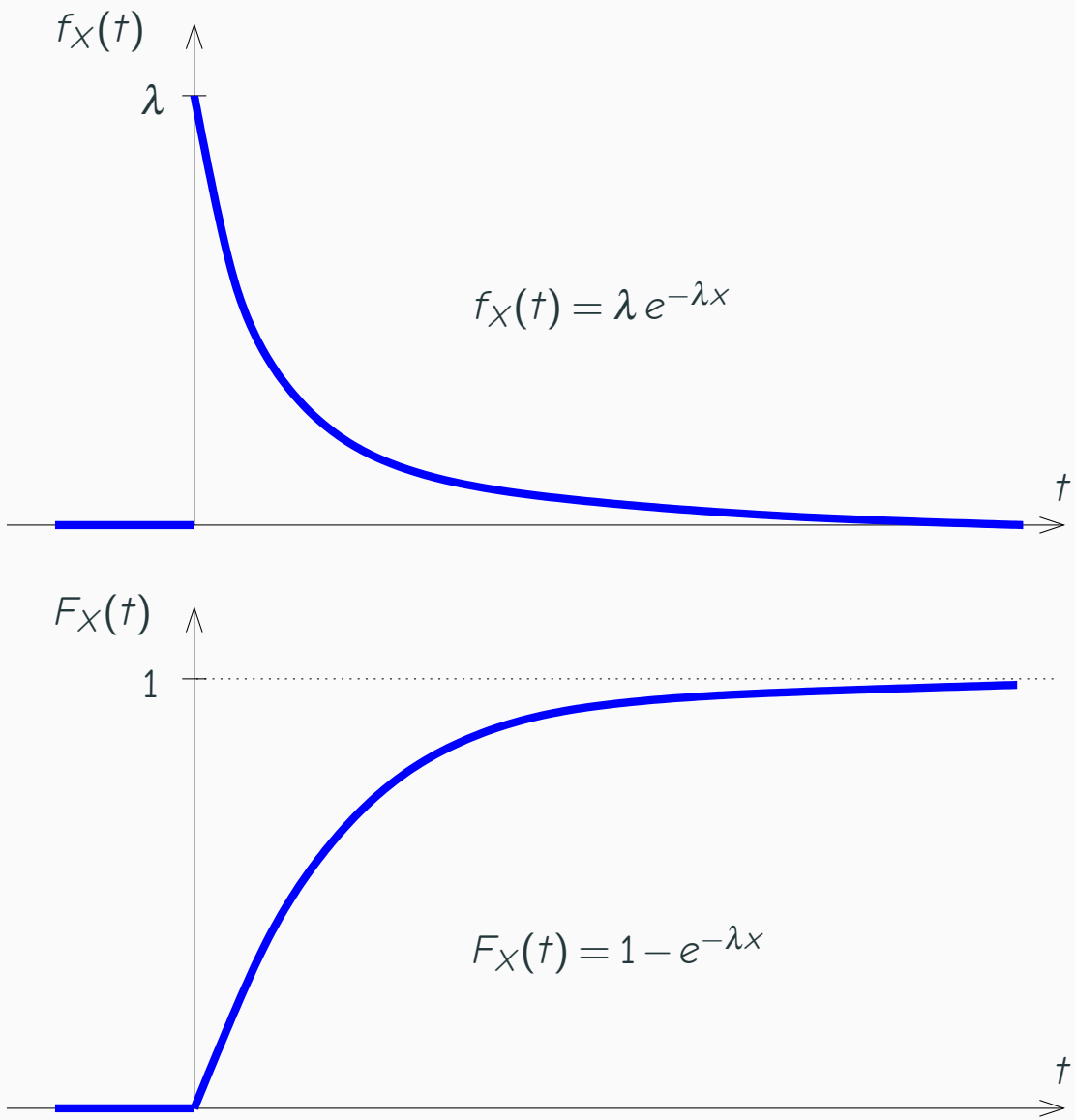
Algoritmo:

1.generar U de $\text{Unif}(0,1)$

2.devolver $X = F_X^{-1}(U)$

$$X \sim f_X(x)$$

Generación de una v.a. con Distribución Exponencial



Generación de una v.a. con Distribución Exponencial

EJEMPLO: Se busca generar valores de una v.a. $X \sim \text{Exp}(\lambda)$ mediante el método de la Transformada Inversa.

$$X \sim \text{Exp}(\lambda) \quad \rightarrow \quad f_X(x) = \lambda e^{-\lambda x} \quad F_X(x) = 1 - e^{-\lambda x}$$

Se cuenta con un algoritmo capaz de generar valores de una v.a. $U \sim \text{Unif}(0,1)$.

$$\begin{aligned} U &= 1 - e^{-\lambda x} = F_X(x) \\ e^{-\lambda x} &= 1 - U \\ \lambda x &= -\log(1-U) \\ x &= -1/\lambda \log(1-U) = F_X^{-1}(U) \end{aligned}$$

Dado que $(1-U)$ también es uniforme en $(0,1)$, resulta más práctico generar X directamente como $-1/\lambda \log U$.

Algoritmo:

1.generar U de $\text{Unif}(0,1)$

2.devolver $X = -1/\lambda \log U$

$$X \sim \text{Exp}(\lambda)$$

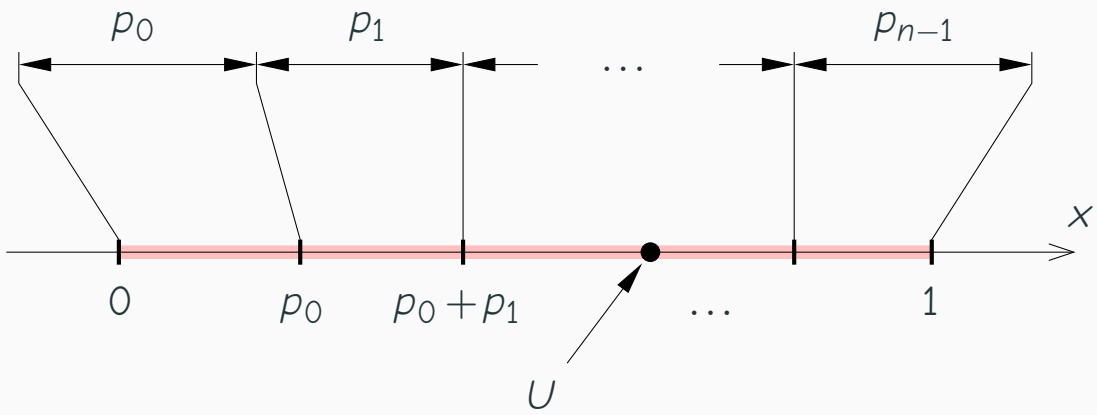
□

Método de la Transformada Inversa — X discreta (I)

Éste es, probablemente, el mecanismo más elemental y directo para generar una v.a.

Siendo $X \sim \mathbb{P}\{X = x_i\} = p_i,\ i = 0,\ldots,n-1,\ \sum_{i=0}^{n-1} p_i = 1$, se sortea $U \sim \text{Unif}(0,1)$ y:

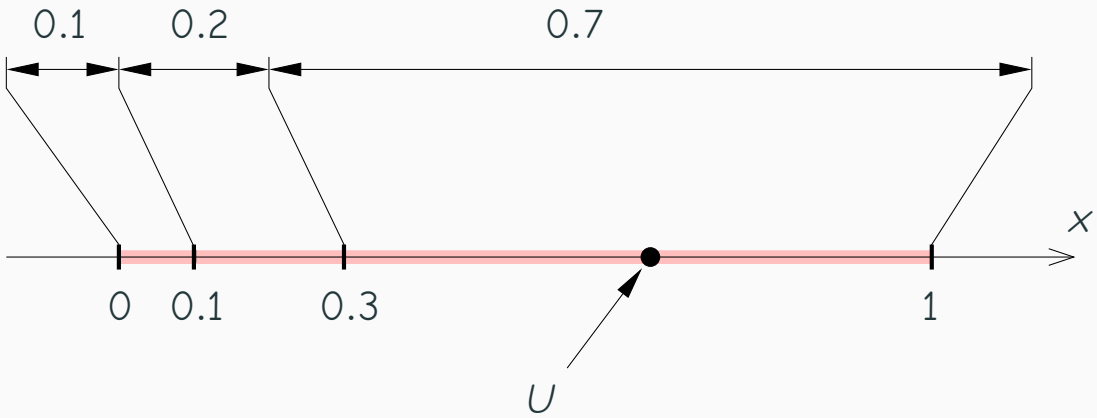
$$
\begin{array}{lccll}
\text{Si} & 0 \leq U < p_0, & X = x_0 & \leftarrow & \text{ocurre con probabilidad } p_0, \\
\text{Si} & p_0 \leq U < p_0 + p_1, & X = x_1 & \leftarrow & \text{ocurre con probabilidad } p_1, \\
\vdots & \vdots & \vdots & & \vdots \\
\text{Si} & p_0 + \ldots + p_{n-2} \leq U < 1, & X = x_{n-1} & \leftarrow & \text{ocurre con probabilidad } p_{n-1}.
\end{array}
$$



EJEMPLO:

$$\text{Generar} \begin{cases} x_0 & \text{c.p. } 0.1 \\ x_1 & \text{c.p. } 0.2 \\ x_2 & \text{c.p. } 0.7 \end{cases}$$

Siendo U uniformemente distribuida en $(0,1)$,

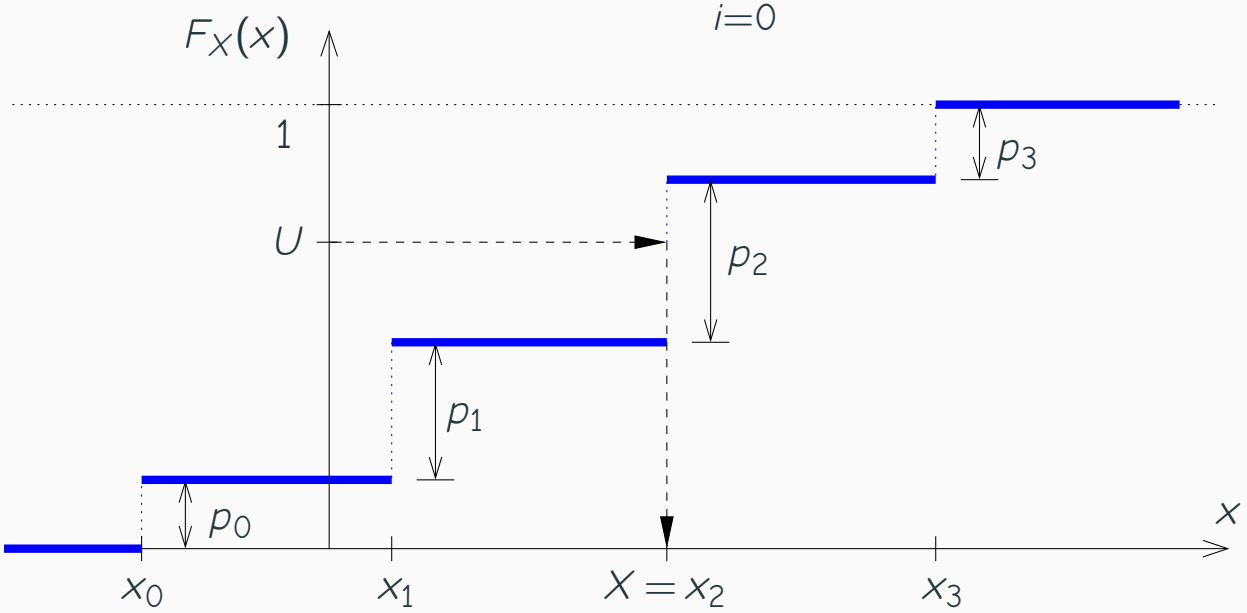


- U cae entre 0 y 0.1 con probabilidad 0.1 $\rightarrow$ x_0
- U cae entre 0.1 y 0.3 con probabilidad 0.2 $\rightarrow$ x_1
- U cae entre 0.3 y 1 con probabilidad 0.7 $\rightarrow$ x_2

El ancho de cada sector determina la probabilidad con la que una U cae dentro de él.

El siguiente ejemplo muestra que el método de la página anterior es numéricamente equivalente al método de la Transformada Inversa.

EJEMPLO: $X \sim \mathbb{P}\{X = x_i\} = p_i,\ i = 0,1,\ldots,3 \qquad \sum_{i=0}^{3} p_i = 1 \qquad x_0 < x_1 < x_2 < x_3$



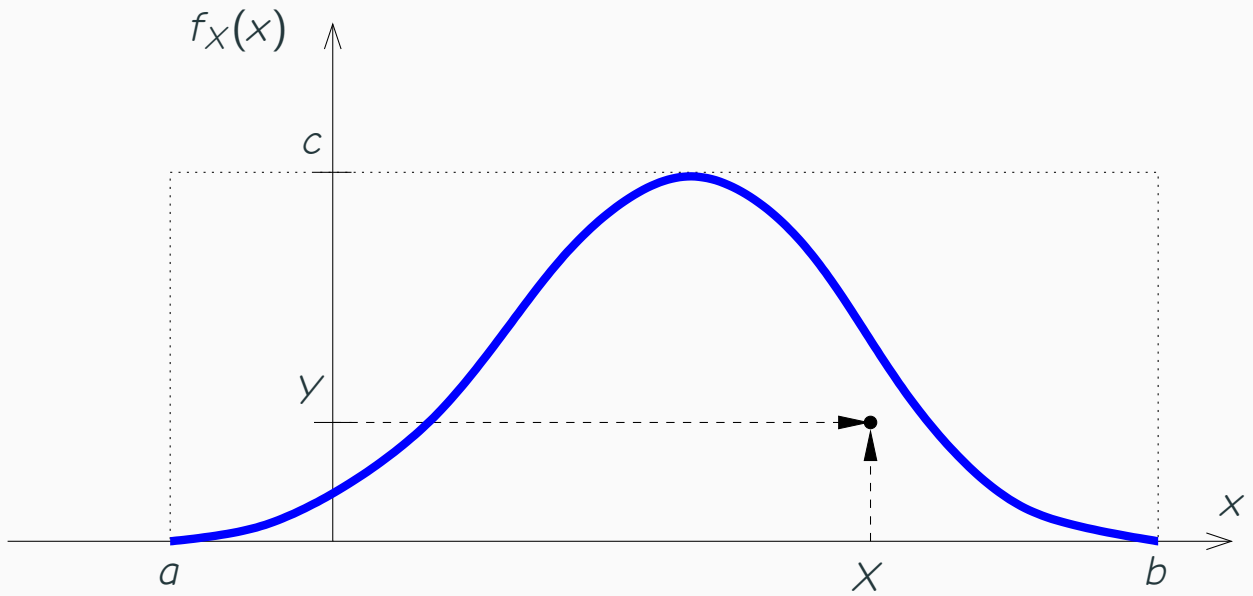
□

Algoritmo:

1. **generar** U de Unif$(0,1)$
2. Encontrar el menor i tal que $U \leq F(x_i)$ y **devolver** $X = x_i$

Método de Aceptación/Rechazo — X continua (I)

Sea $f_X(x) = 0$ si $x \leq a$ y $x \geq b$, y sea $c = \sup\{\, f_X(x) : x \in [a,b] \,\}$



Algoritmo:

1. **generar** X de Unif(a,b)
2. **generar** Y de Unif$(0,c)$
3. **si** $Y \leq f_X(X)$ **devolver** X, **si no**, **volver a 1**.

$$X \sim f_X(x)$$

Bibliografía


S. M. Ross. *Simulation, Fourth Edition*. Academic Press, Inc., 2006. ISBN: 0-12-598063-9.


S.M. Ross. *Introduction to Probability Models*. 10th ed. Elsevier Science, 2006. ISBN: 9780123756879.


R.Y. Rubinstein and D.P. Kroese. *Simulation and the Monte Carlo Method*. Wiley, 2008. ISBN: 9780470230374.