

# Métodos Computacionales

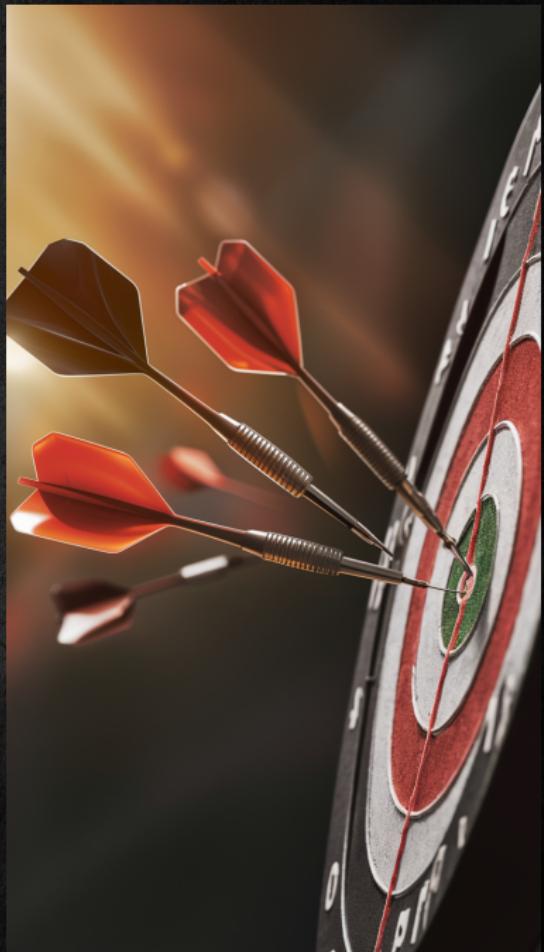
## Montecarlo

Octubre 9, 2024

S. Alexis Paz



Departamento de  
QUÍMICA TEÓRICA  
Y COMPUTACIONAL  
Facultad de Ciencias Químicas  
Universidad Nacional de Córdoba



hecho con ideogram

STEVEN E. KONIN / DAWN C. MEREDITH

# COMPUTATION PHYSICS

FORTRAN VERSION

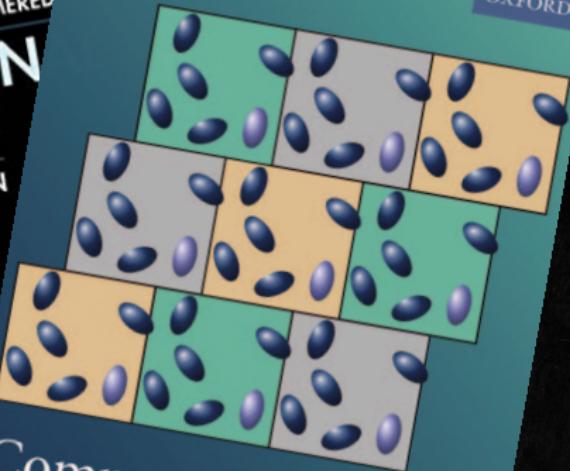


## Computer Simulation of Liquids

SECOND EDITION

Michael P. Allen & Dominic J. Tildesley

OXFORD



# Variables Aleatorias

Una variable aleatoria es una función que asigna un valor a cada resultado posible de un experimento aleatorio.

cara → 1

cruz → -1

espacio muestral



Se define en términos de los valores que puede tomar ( $x \in \Omega$ ) y la probabilidad de cada uno de esos valores.

Caso Discreto →  $X, P(x)$

Caso Contínuo →  $X, \rho(x)$

# Variables Aleatorias

Caso Discreto  $\rightarrow X, P(x)$

$$\sum_i^N P(x_i) = 1$$

Caso Continuo  $\rightarrow X, \rho(x)$

$$\int_{\Omega} \rho(x) dx = 1$$

$$P[a \leq X \leq b] = \int_a^b \rho(x) dx$$

$N$  es el cardinal de  $\Omega$

# Valor de Expectación

Caso Discreto  $\rightarrow X, P(x)$

$$\langle X \rangle = \sum_i^N x_i P(x_i)$$

$$\langle f(X) \rangle = \sum_i^N f(x_i) P(x_i)$$

Caso Continuo  $\rightarrow X, \rho(x)$

$$\langle X \rangle = \int_{\Omega} x \rho(x) \, dx$$

$$\langle f(X) \rangle = \int_{\Omega} f(x) \rho(x) \, dx$$

Varianza:  $\sigma^2(X) = \langle X^2 \rangle - \langle X \rangle^2$

# Valor de Expectación

*M* es el número de observaciones

Estimador:  $\overline{f(X)} = \frac{1}{M} \sum_i^M f(X_i)$        $\lim_{M \rightarrow \infty} \overline{f(X)} = \langle f(X) \rangle$

Demostración:

Caso Continuo

$$\begin{aligned} P(X_i) &\approx P(x_i \leq X \leq x_i + h) \\ &\approx \rho(x_i)h \end{aligned}$$

$$\begin{aligned} &\approx \sum_i^N f(x_i) \rho(x_i)h \\ &\approx \int_a^b f(x_i) \rho(x_i) dx \end{aligned}$$

Caso  
Discreto  


$$\begin{aligned} \overline{f(X)} &= \frac{1}{M} \sum_i^M f(X_i) \\ &= \sum_i^N f(x_i) \frac{N_i}{M} \\ &\approx \sum_i^N f(x_i) P(X = x_i) \\ &= \langle f(X) \rangle \end{aligned}$$

$$\left\langle \frac{f(X)}{\rho(X)} \right\rangle = \int_{\Omega} f(x) \, dx \approx \overline{\left( \frac{f(X)}{\rho(X)} \right)}$$

¡Es un método para resolver integrales!

$$\left\langle \frac{f(X)}{\rho(X)} \right\rangle = \int_{\Omega} f(x) dx \approx \overline{\left( \frac{f(X)}{\rho(X)} \right)}$$

Uso variables aleatorias → MONTE CARLO.

Pero... ¿Qué error tiene?

## Teorema central del límite

Sea  $Y = \sum_i^L X^{(i)}$  donde  $X^{(i)}$  son variables independientes e identicamente distribuidas, tal que  $\langle Y \rangle = L\langle X \rangle$  y  $\sigma^2(Y) = L\sigma^2(X)$ . Entonces la distribución de  $Y$  converge a la normal cuando  $L \rightarrow \infty$ .

Si bien usamos  $X_j^{(i)}$  para indicar un resultado posible al observar  $X^{(i)}$

Notese que  $Y_j = \sum_i^L X_j^{(i)} = \sum_i^L X_i^{(j)}$

Si repetimos este método usando diferentes  $\{X_1^{(i)}, X_2^{(i)}, \dots, X_M^{(i)}\}$

tendremos diferentes valores aleatorios para  $Y_j = \sum_i^M \frac{1}{M} \frac{f(X_i^{(j)})}{\rho(X_i)}$

Entonces

$$\langle Y \rangle = M \left\langle \frac{1}{M} \frac{f(X)}{\rho(X)} \right\rangle = \left\langle \frac{f(X)}{\rho(X)} \right\rangle = \int f(x) dx$$

y además

$$\sigma(Y) = \sqrt{M} \sigma \left( \frac{1}{M} \frac{f(X)}{\rho(X)} \right) = \sigma \left( \frac{f(X)}{\rho(X)} \right) \frac{1}{\sqrt{M}}$$

y como  $Y$  sigue una distribución normal, se puede esperar que

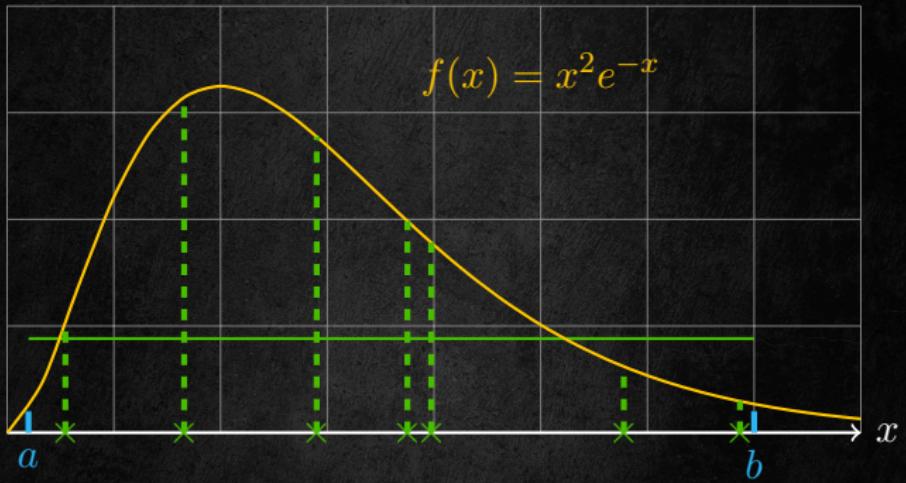
$$\mathcal{O}(Y_j - \langle Y \rangle) \approx \sigma(Y) = \mathcal{O}\left(\frac{1}{\sqrt{M}}\right)$$

# Monte Carlo

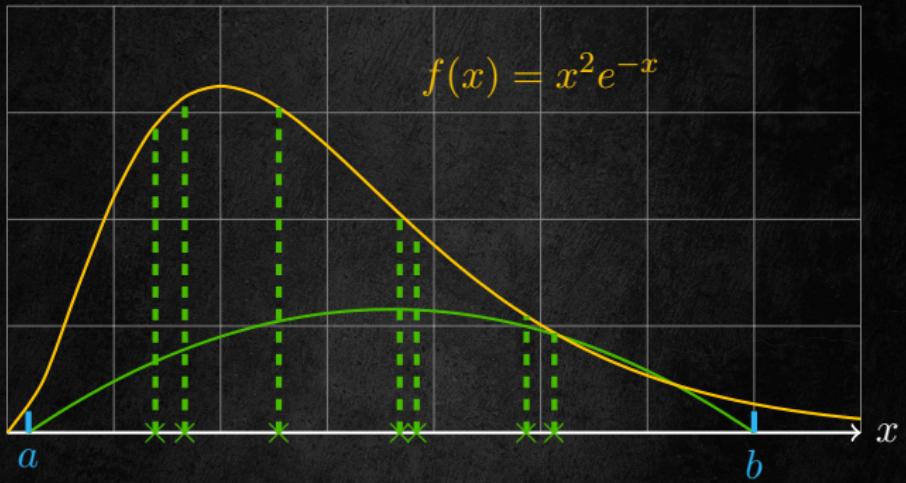
$$\int_{\Omega} f(x) dx \approx \overline{\left( \frac{f(X)}{\rho(X)} \right)} \quad \text{con } \mathcal{O}\left(\frac{1}{\sqrt{M}}\right)$$

Uso  $X$  con cualquier distribución  $\rho(x)$ .

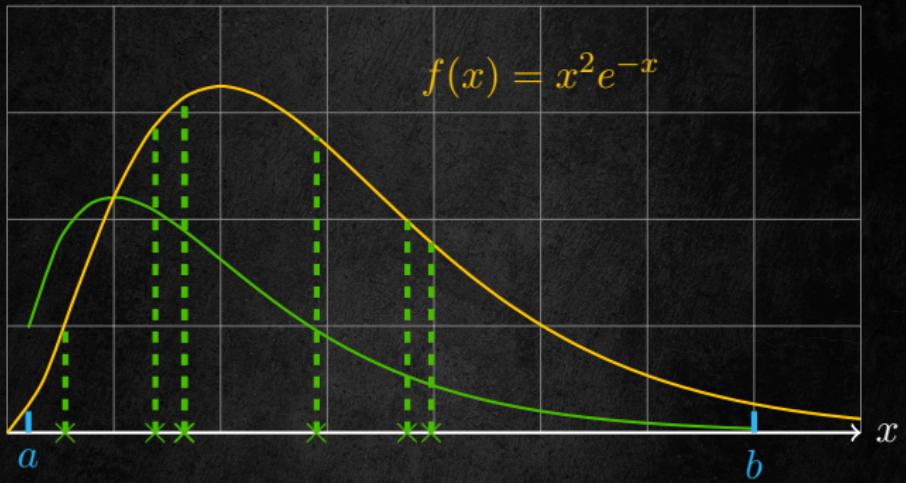
Pero... ¿Conviene alguna?



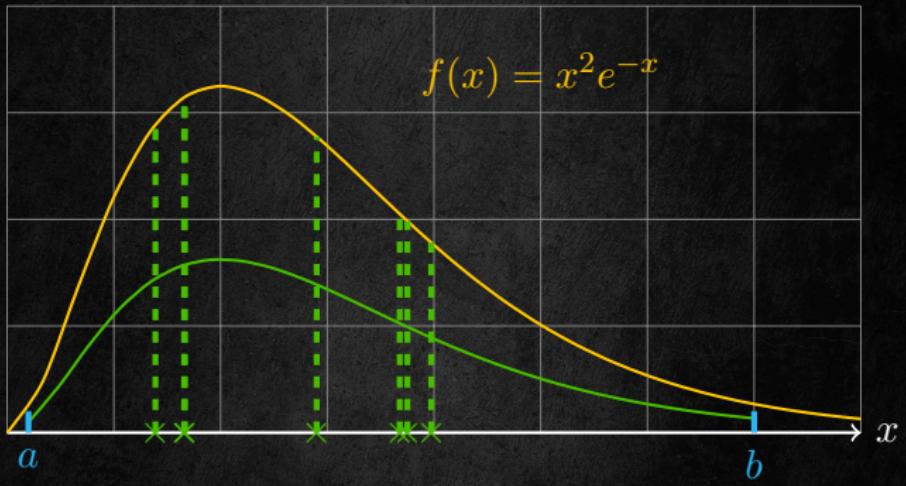
Si  $\rho(x) = \frac{1}{b-a}$ ,  $\int_a^b f(x) dx \approx (b-a) \overline{X^2 e^{-X}}$



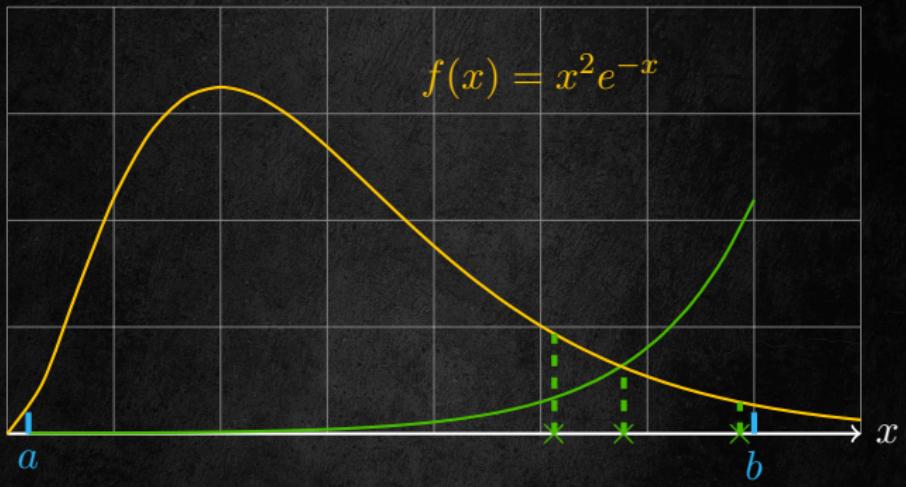
Si  $\rho(x) = -c(x - a)(x - b)$ ,  $\int_a^b f(x) dx \sim -\overline{e^{-X}}/c$



$$\text{Si } \rho(x) = cxe^{-x}, \quad \int_a^b f(x) dx \sim \overline{X}/c$$



Si  $\rho(x) = cx^2 e^{-x}$ ,  $\int_a^b f(x) dx \sim \bar{1}/c$



Si  $\rho(x) = ce^x$ ,  $\int_a^b f(x) dx \sim \overline{X^2 e^{-2X}/c}$

# Muestreo de Importancia

$$\int_{\Omega} f(x) dx \approx \overline{\left( \frac{f(X)}{\rho(X)} \right)} \quad \text{con} \quad \mathcal{O}\left(\frac{1}{\sqrt{N}}\right)$$

Uso  $X$  con  $\rho(x) \sim f(x)$ .

Pero... ¿Como genero  $X$ ?

# Generador de números pseudoaleatorios

| Algoritmo                        | Año  | Autores                                     | Referencias        | Descripción   |
|----------------------------------|------|---|--------------------|---|
| Cuadrado Medio                   | 1946 | John von Neumann                            | <a href="#">3</a>  | Un PRNG considerado de baja calidad pero de gran relevancia histórica por ser uno de los algoritmos pioneros.   |
| Generador de Lehmer              | 1951 | D. H. Lehmer                                | <a href="#">4</a>  | También conocido como método Congruente Lineal Multiplicativo y de gran influencia en este campo de estudio.  |
| Generador lineal congruencial    | 1958 | W. E. Thomson                               | <a href="#">5</a>  | Modelo derivado de Lehmer (1951) de gran influencia y muy estudiado en todo el mundo.   |
| Generador Lagged Fibonacci (LFG) | 1958 | G. J. Mitchell; D.P. Moore                  | <a href="#">6</a>  | Un algoritmo muy influyente en el campo del estudio de los procesos de generación de números aleatorios que inspiró a otros grandes autores en los siguientes años como George Marsaglia, creador del valorado test de calidad de números aleatorios llamado "Diehard", por ejemplo.            |
| Linear-feedback Shift Register   | 1965 | R. C. Tausworthe                            | <a href="#">7</a>  | Un generador cuyo diseño influyó en muchos otros PRNG posteriores. Por lo tanto, es muy importante desde el punto de vista histórico. También conocido como el generador Tausworthe.  |
| Generador de Wichmann-Hill       | 1982 | B. A. Wichmann; D. I. Hill                  | <a href="#">8</a>  | Una combinación de tres pequeños LCGs, adecuados para CPUs de 16 bits. Ampliamente utilizado en muchos programas, por ejemplo se utilizó en Excel 2003 y algunas versiones posteriores para la función RAND de Excel y fue el generador por defecto en el lenguaje Python hasta la versión 2.2. |
| Rule 30                          | 1983 | Stephen Wolfram                             | <a href="#">9</a>  | Generador basado en autómatas celulares.  |
| Blum Blum Shub                   | 1986 | Manuel Blum; Leonore Blum; Michael Shub     | <a href="#">10</a> | Considerado uno de los generadores más seguros desde el punto de vista criptográfico, debido principalmente a la implementación en su fórmula de estudios y conceptos derivados de la teoría de números.  |
| Generador de Park-Miller         | 1988 | S. K. Park; K. W. Miller                    | <a href="#">11</a> | Una implementación específica de un generador de Lehmer, ampliamente utilizada porque se incluye en C++ como la función <code>minstd_rand0</code> a partir de C++11.  |
| MIXMAX                           | 1991 | G. K. Savvidy; N. G. Ter-Arutyunyan-Savvidy | <a href="#">12</a> | Es un generador que pertenece a la clase de generador lineal congruente matricial, una generalización del Método Congruente Lineal. La lógica de la familia de generadores MIXMAX se basa en los resultados de la teoría ergódica y la mecánica clásica.  |
| Add-with-carry                   |      | G. Marsaglia; A.                            | <a href="#">13</a> |   |

| Evolution/ Soccer (P.E.S.)                     |      |  |                    |   |
|--|------|--|--------------------|---|
| Xorshift                                       | 2003 | G. Marsaglia                           | <a href="#">16</a> | Es un subtipo muy rápido de generadores LFSR. Marsaglia también propuso como mejora el generador xorwow, en el que la salida de un generador xorshift se suma con una secuencia de Weyl. El generador xorwow es el generador por defecto de la biblioteca CURAND de la interfaz de programación de aplicaciones nVidia CUDA para unidades de procesamiento gráfico. |
| Fortuna  | 2003 | Bruce Schneier;<br>Niels Ferguson      |                    | Algoritmo considerado criptográficamente seguro. Un CSPRNG muy conocido por ser implementado en los sistemas y productos de Apple.  |
| Well equidistributed long-period linear (WELL) | 2006 | F. Panneton; P. L'Ecuyer; M. Matsumoto | <a href="#">17</a> | Algoritmo conocido por ser complementario al Mersenne Twister (MT), buscando deliberadamente cubrir sus puntos débiles.   |
| Advanced Randomization System (ARS)            | 2011 | J. Salmon; M. Moraes; R. Dror; D. Shaw | <a href="#">18</a> | Una versión simplificada del cifrado en bloque AES, que permite un rendimiento muy rápido en el sistema que soporta AES-NI.   |
| Permuted Congruential Generator (PCG)          | 2014 | M. E. O'Neill                          | <a href="#">19</a> | Un modelo derivado del Método Lineal Congruencial.  |
| Random Cycle Bit Generator (RCB)               | 2016 | R. Cookman                             | <a href="#">20</a> | El RCB se describe como un generador de patrones de bits hecho para superar algunas de las deficiencias con el Mersenne Twister (MT) y la restricción de duración de período/bit corto de los generadores de desplazamiento/módulo.   |
| Xoroshiro128+                                  | 2018 | D. Blackman; S. Vigna                  | <a href="#">21</a> | Una modificación de los generadores Xorshift de G. Marsaglia, uno de los generadores más rápidos en las CPUs modernas de 64 bits. Los generadores relacionados son xoroshiro128**, xoroshiro256+ y xoroshiro256***.   |
| 64-bit MELG (MELG-64)                          | 2018 | S. Harase; T. Kimoto                   | <a href="#">22</a> | Una implementación de generadores lineales F2 de 64 bits con el período primario de Mersenne.   |
| Squares RNG                                    | 2020 | B. Widynski                            | <a href="#">23</a> | Generador derivado del método del cuadrado medio propuesto por Jhon von Neumann.  |
| Itamaracá (Ita)                                | 2021 | D. H. Pereira                          | <a href="#">24</a> | Conocido por ser el primer algoritmo PRNG que tiene la Función de Valor Absoluto en su base. Itamaracá también se presenta como un modelo sencillo y rápido que genera secuencias de números aleatorios aperiódicos.  |

# Generador lineal congruencial

$$X_i = (aX_{i-1} + c) \bmod m$$

(la operación módulo obtiene el resto de la división)

|          |          |          |          |          |          |          |          |          |          |
|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| $m = 9$  | 8 0<br>7 |
| $a = 2$  | 2        | 4        | 6        | 8        | 0        | 2        | 4        | 6        | 8        |
| $c = 0$  | 3        | 5        | 6        | 7        | 1        | 3        | 5        | 6        | 7        |
| seed = 1 | output 2 | output 4 | output 8 | output 7 | output 5 | output 5 | output 1 | output 1 | output 2 |

|          |          |          |          |
|----------|----------|----------|----------|
| $m = 9$  | 8 0<br>7 | 8 0<br>7 | 8 0<br>7 |
| $a = 2$  | 2        | 4        | 6        |
| $c = 0$  | 3        | 5        | 6        |
| seed = 3 | output 6 | output 3 |          |

|          |          |          |          |          |          |          |          |          |          |
|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| $m = 9$  | 8 0<br>7 |
| $a = 4$  | 2        | 4        | 6        | 8        | 0        | 2        | 4        | 6        | 8        |
| $c = 1$  | 3        | 5        | 6        | 7        | 1        | 3        | 5        | 6        | 7        |
| seed = 0 | output 1 | output 5 | output 3 | output 4 | output 8 | output 6 | output 7 | output 2 | output 0 |

# Teorema de transformación de variables aleatorias

Supongamos que sabemos generar  $X$  con distribución  $\rho(x)$ ,

la variable  $Z = f(X)$  tendrá alguna distribución  $q(x) \dots$

¿Qué función  $f(x)$  permitirá obtener una determinada  $q(x)$ ?

$$q(z) = \int_{\Omega} \delta(f(x) - z) \rho(x) dx$$

si existe  $f^{-1}(x)$

$$q(z) = \rho(f^{-1}(z)) \frac{df^{-1}(z)}{dz}$$

si  $\rho(x) = 1$  en  $(0, 1)$

$$f^{-1}(z) = \int q(z) dz + C$$

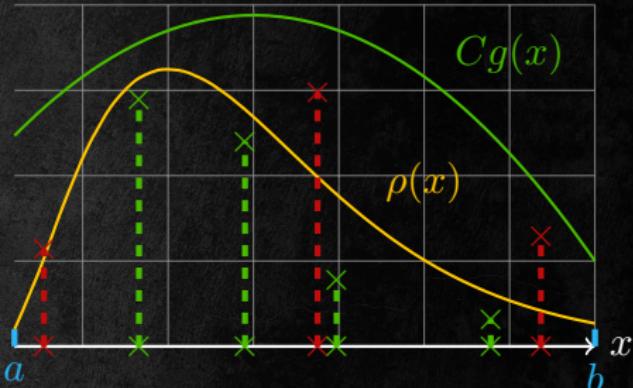
# Método del rechazo de Von Neumann

1. Elijo cualquier  $Cg(x) \geq \rho(x)$

2. Genero pares  $(X, Y)$  uniformes en el área debajo de  $Cg(x)$ . Para ello:

a. genero  $X$  con distribución  $g(x)$ .

b. para cada  $X$  genero  $Y$  uniforme en  $[0, Cg(X)]$ .



3. **Rechazo/ignoro** aquellos  $X$  que caen por arriba de  $\rho(x)$

4. Los restantes  $X$  tendrán distribución  $\rho(x)$ .

## Buffon's Needle Experiment

