

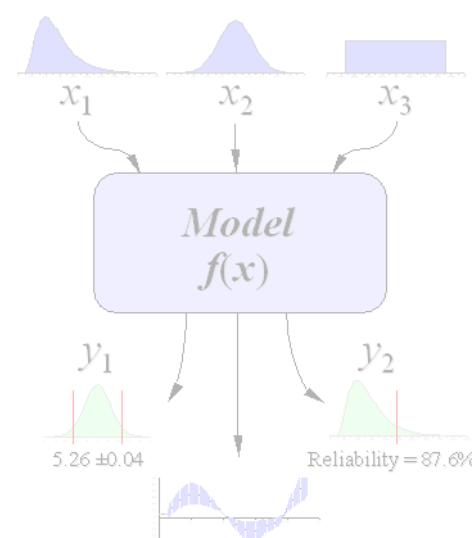
TEMA 5

NÚMEROS ALEATORIOS Y MÉTODOS DE MONTE CARLO

(1 sesión teórica + 2 sesiones prácticas)



Rocío del Río (rrio@us.es)
Dpto. Electrónica y Electromagnetismo





Contenidos

- Funciones de MATLAB relacionadas con la aleatoriedad:
`rand`, `randn`, `randi`, `random` y `randperm`
- Generadores de números pseudo-aleatorios:
Concepto, tests de caracterización y repetitividad
- Métodos de Monte Carlo y aplicaciones:
 - Paseo aleatorio
 - Desintegración radioactiva
 - Integración numérica

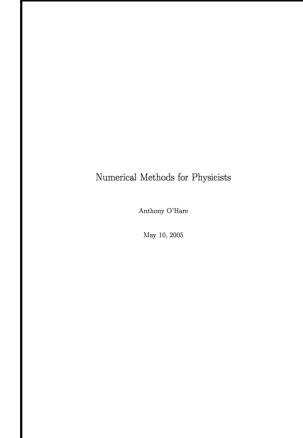
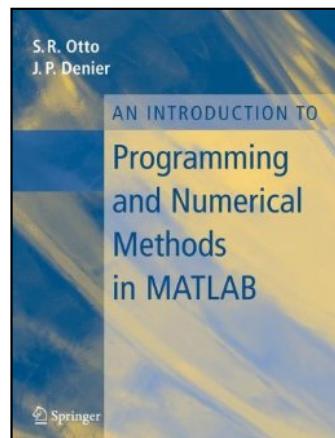
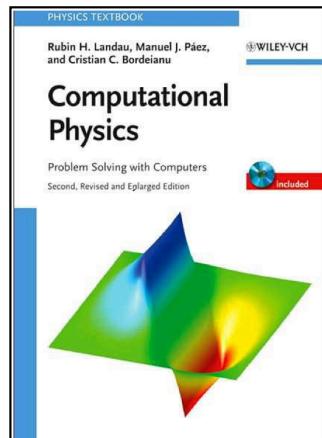


Objetivos

- Saber utilizar las funciones `rand`, `randi` y `randn` de MATLAB para la generación de secuencias aleatorias.
- Comprender la naturaleza pseudo-aleatoria de cualquier secuencia generada de manera determinista por un ordenador.
- Ser capaz de aplicar tests simples para determinar la uniformidad y aleatoriedad de secuencias pseudo-aleatorias.
- Comprender la utilidad de los métodos de Monte Carlo en la simulación de procesos físicos estocásticos.
- Ser capaz de evaluar integrales multi-dimensionales mediante el muestreo del valor medio.

Bibliografía

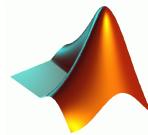
- R.H. Landau, M.J. Páez, C.C. Bordeianu: *Computational Physics: Problem Solving with Computers, 2/E.* Wiley, 2007.
<http://physics.orst.edu/~rubin/> → Transparencias, videos, applets, ...
- S.R. Otto, J.P. Denier: *An Introduction to Programming and Numerical Methods in MATLAB.* Springer, 2005.
- A. O'Hare: *Numerical Methods for Physicists.* 2005.
<http://www-teaching.physics.ox.ac.uk/computing/NumericalMethods/NMfP.pdf>



Función rand

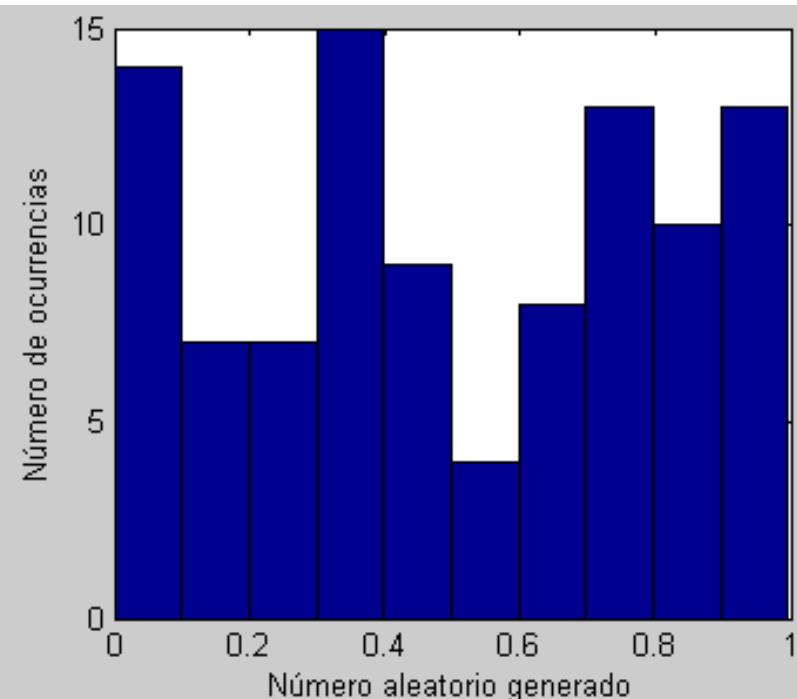
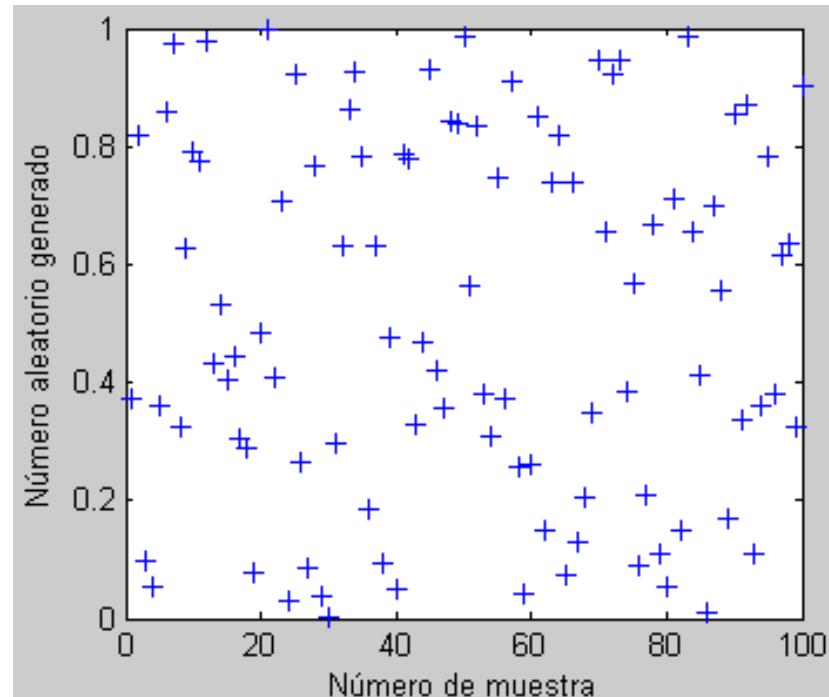
- Genera números reales aleatorios con distribución uniforme en el intervalo [0,1].

rand (M, N) → genera una matriz MxN



EJEMPLO_01.m

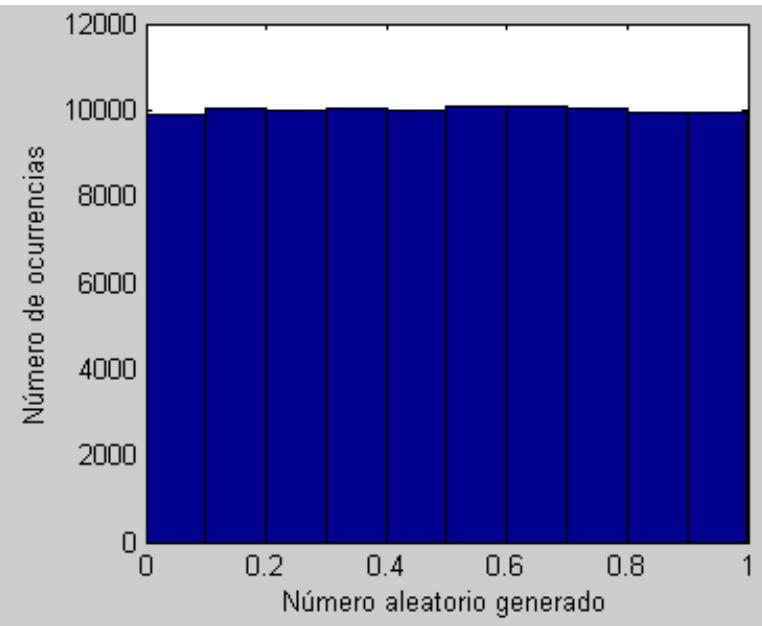
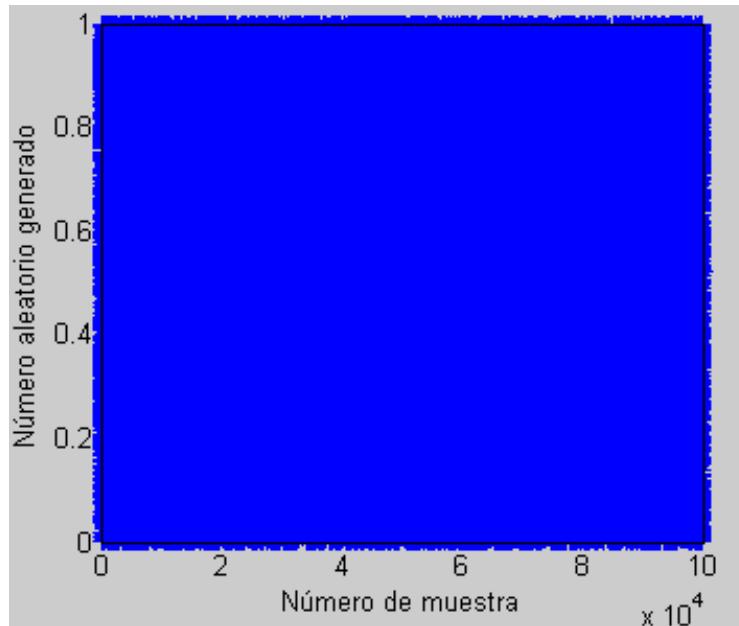
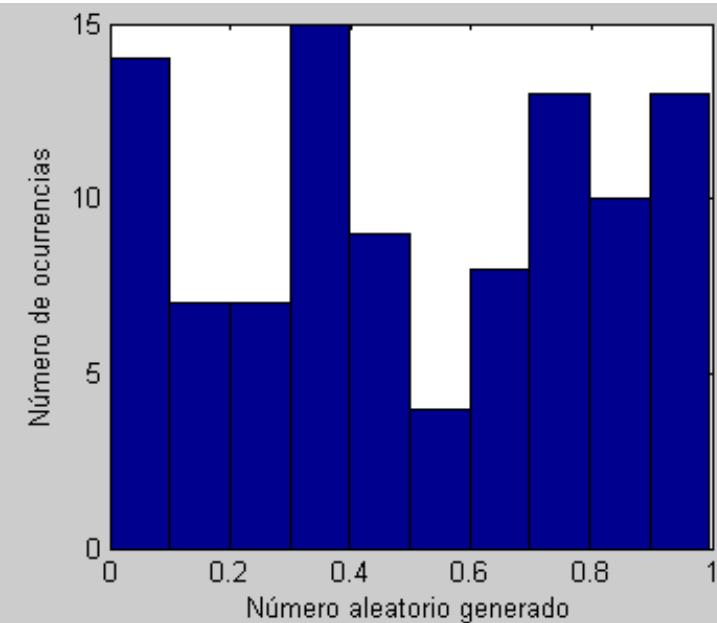
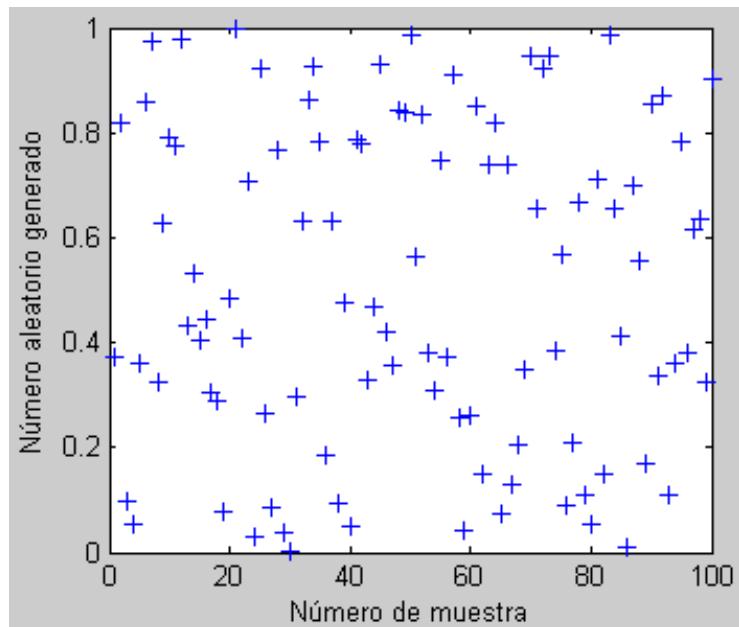
```
>> ...
>> rand(1,100)
>> ...
```



```
>> rand(1,100)
```

La función de distribución es tanto más visible cuanto mayor es el número de muestras generadas.

```
>> rand(1,1e5)
```

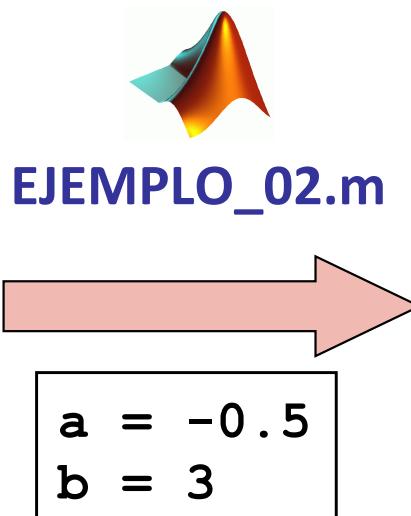
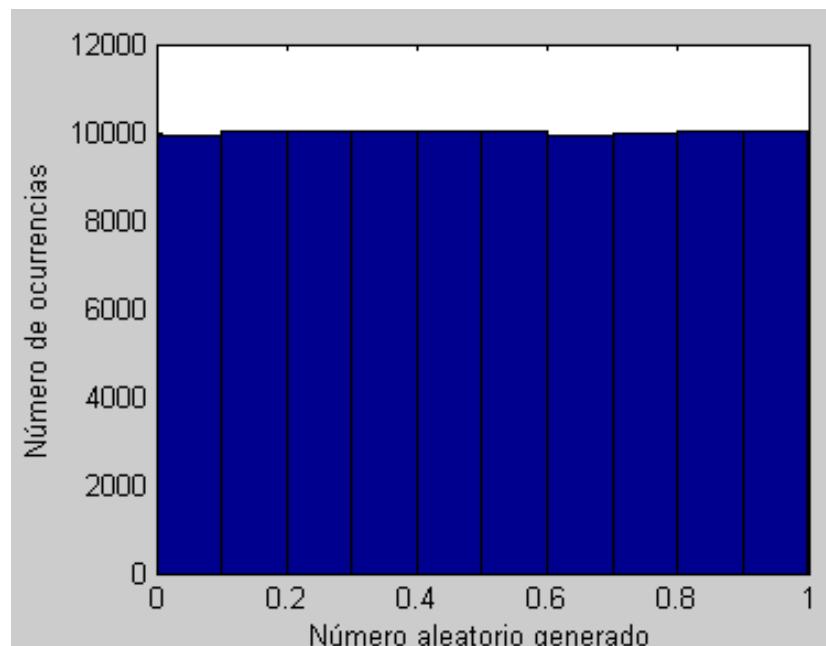




Desplazamiento de Distribuciones Uniformes

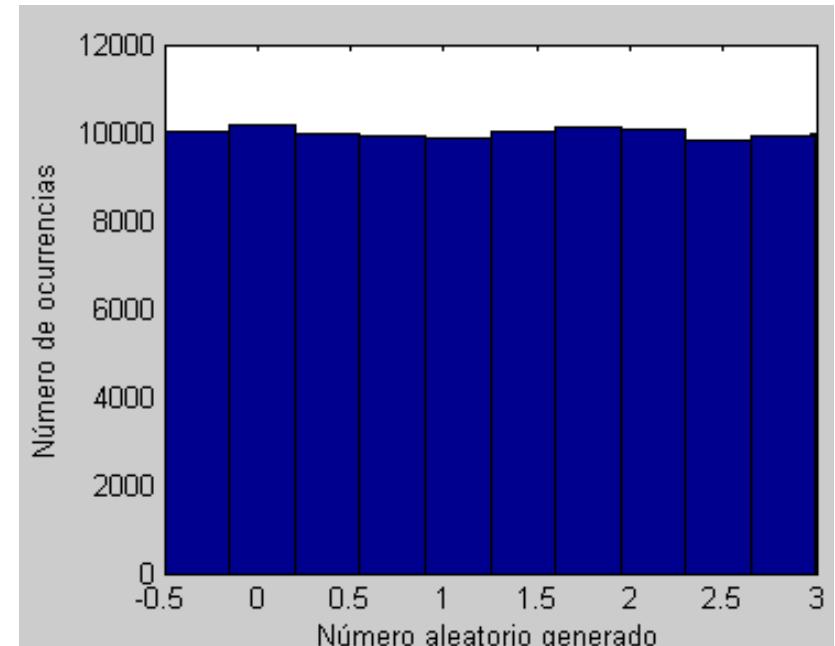
Números reales aleatorios con distribución uniforme en el intervalo **[0,1]**.

```
>> rand(1,1e5)
```



Números reales aleatorios con distribución uniforme en el intervalo **[a,b]**.

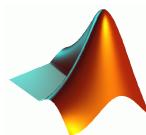
```
>> a+(b-a)*rand(1,1e5)
```



Función randn

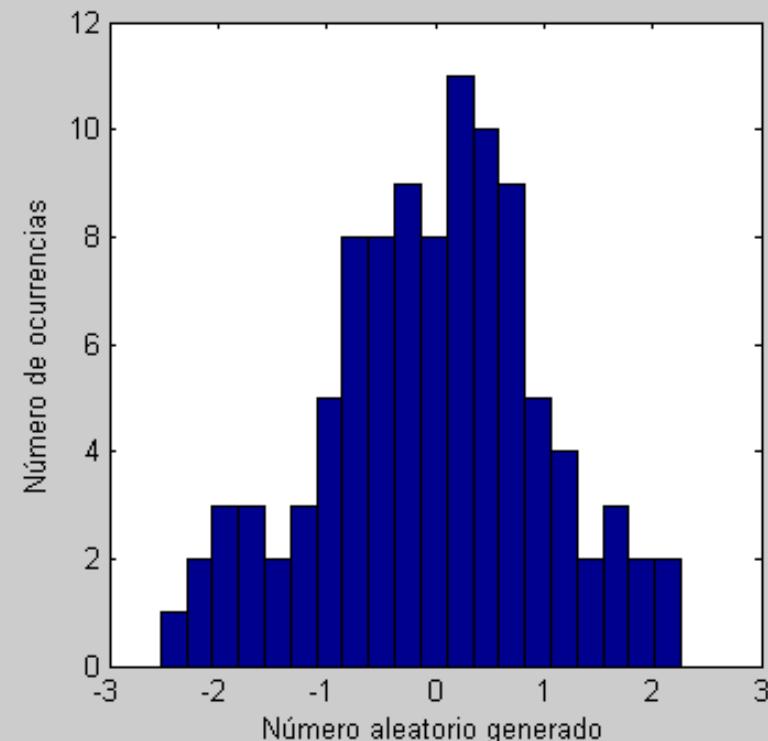
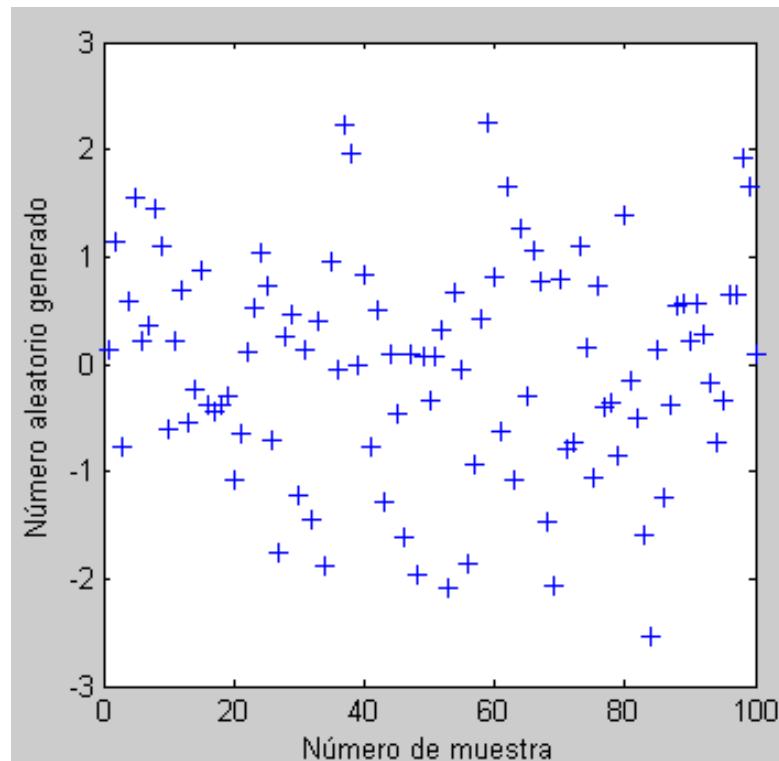
- Genera números reales aleatorios con distribución normal (gaussiana) de media 0 y desviación estándar 1.

randn (M, N) → genera una matriz MxN



EJEMPLO_03.m

```
>> ...
>> randn(1,100)
>> ...
```



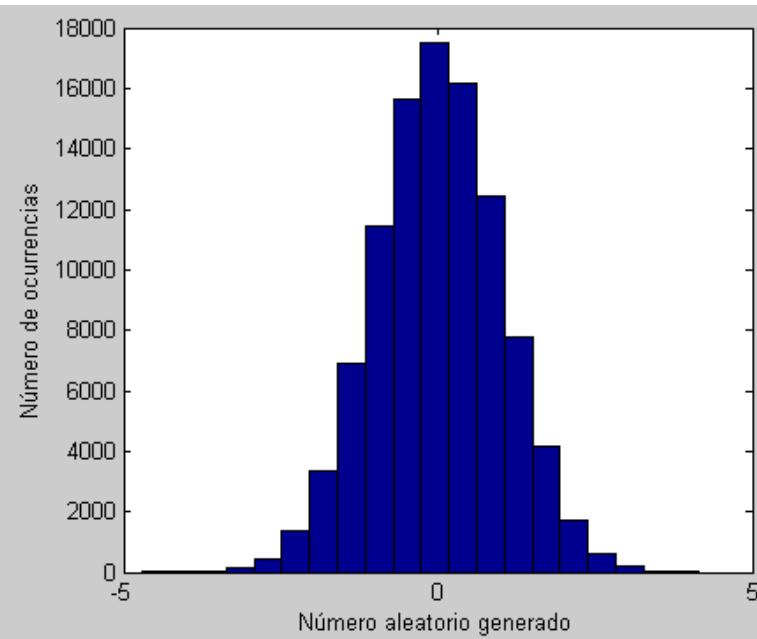
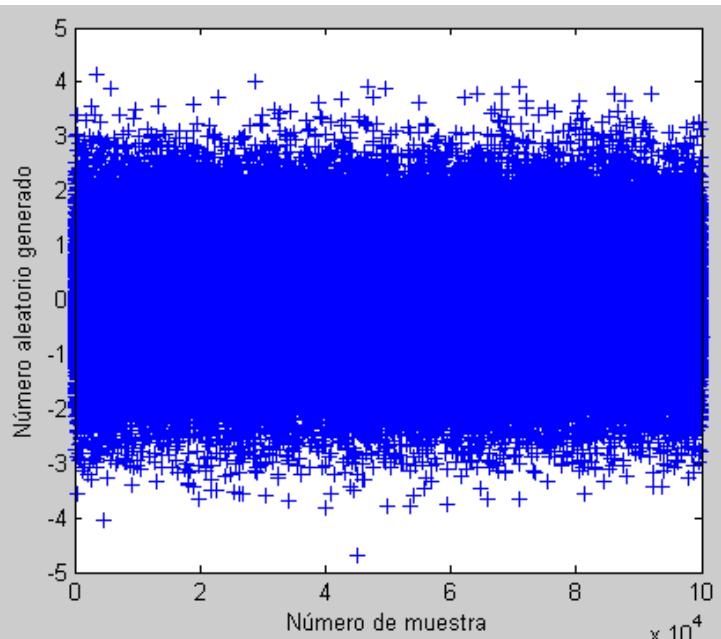
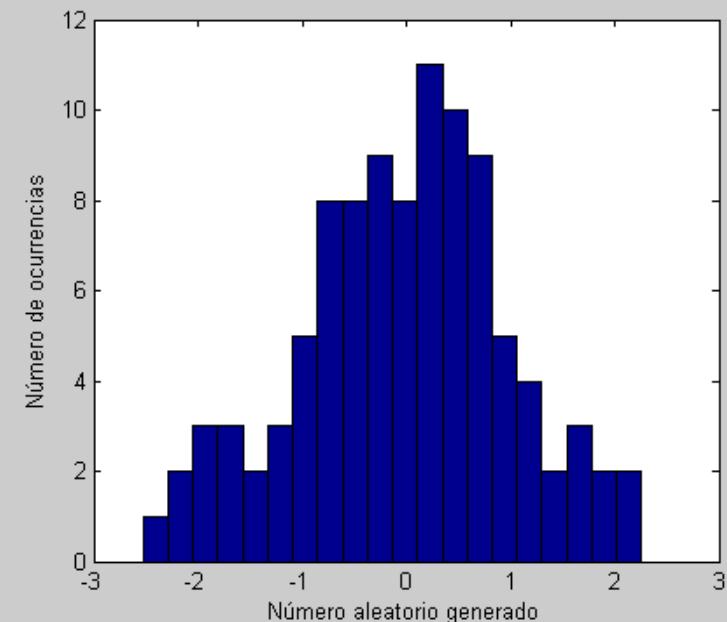
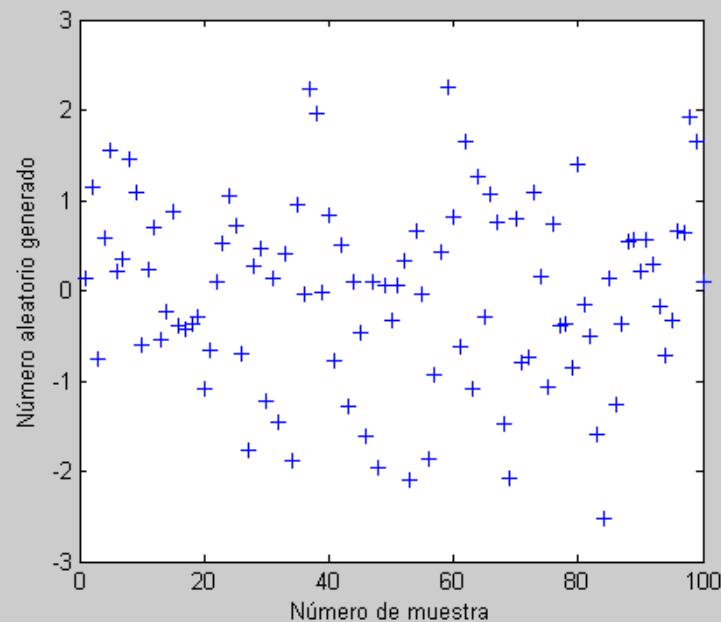
```
>> randn(1,100)
```

Media = -0.0177
Std = 1.0150

La función de distribución es tanto más visible cuanto mayor es el número de muestras generadas.

```
>> randn(1,1e5)
```

Media = 0.0021
Std = 0.9979





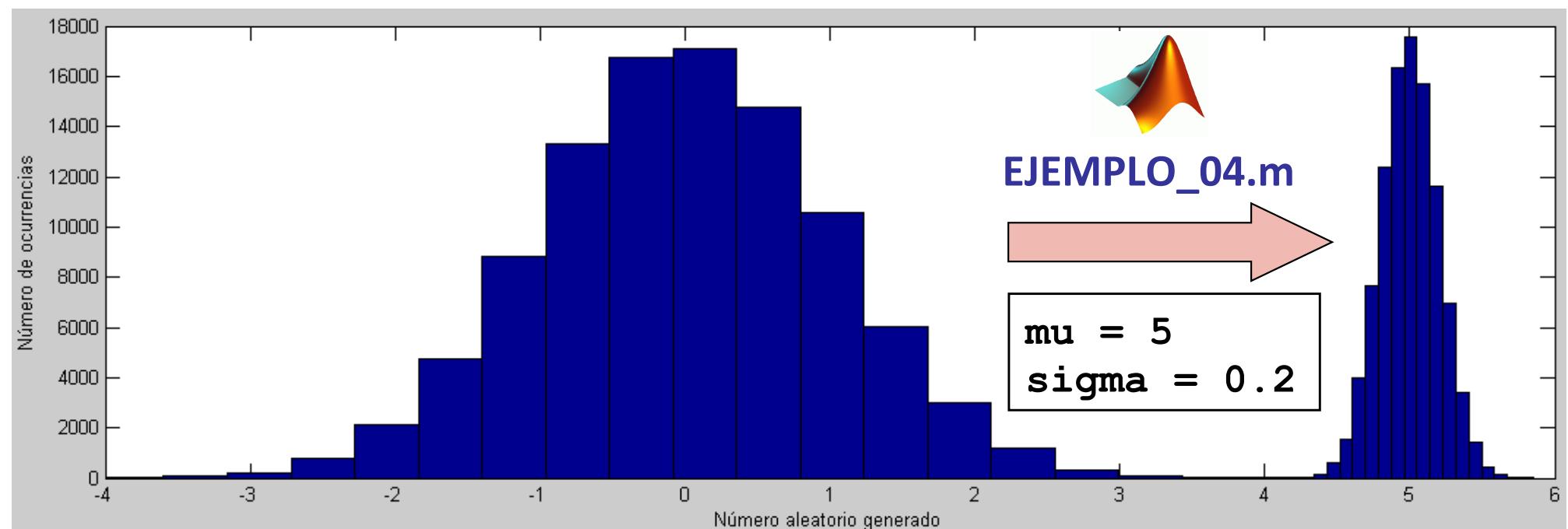
Desplazamiento de Distribuciones Normales

Números reales aleatorios con distribución normal de **media 0** y **desviación estándar 1**.

```
>> randn(1,1e5)
```

Números reales aleatorios con distribución normal de **media μ** y **desviación estándar σ** .

```
>> mu+sigma*randn(1,1e5)
```



Función random

- Genera números reales aleatorios de acuerdo a distribuciones especificadas.

random (NAME ,A ,B ,C ,M ,N) → Matriz MxN con distribución NAME y parámetros A, B, C

```
>> help random
```

...

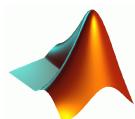
NAME can be:

'beta' or 'Beta',	'nbin' or 'Negative Binomial',
'bino' or 'Binomial',	'ncf' or 'Noncentral F',
'chi2' or 'Chisquare',	'nct' or 'Noncentral t',
'exp' or 'Exponential',	'ncx2' or 'Noncentral Chi-square',
'ev' or 'Extreme Value',	'norm' or 'Normal',
'f' or 'F',	'poiss' or 'Poisson',
'gam' or 'Gamma',	'rayl' or 'Rayleigh',
'gev' or 'Generalized Extreme Value',	't' or 'T',
'gp' or 'Generalized Pareto',	'unif' or 'Uniform',
'geo' or 'Geometric',	'unid' or 'Discrete Uniform',
'hyge' or 'Hypergeometric',	'wbl' or 'Weibull'.
'logn' or 'Lognormal',	
'nbin' or 'Negative Binomial',	

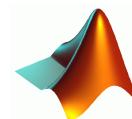
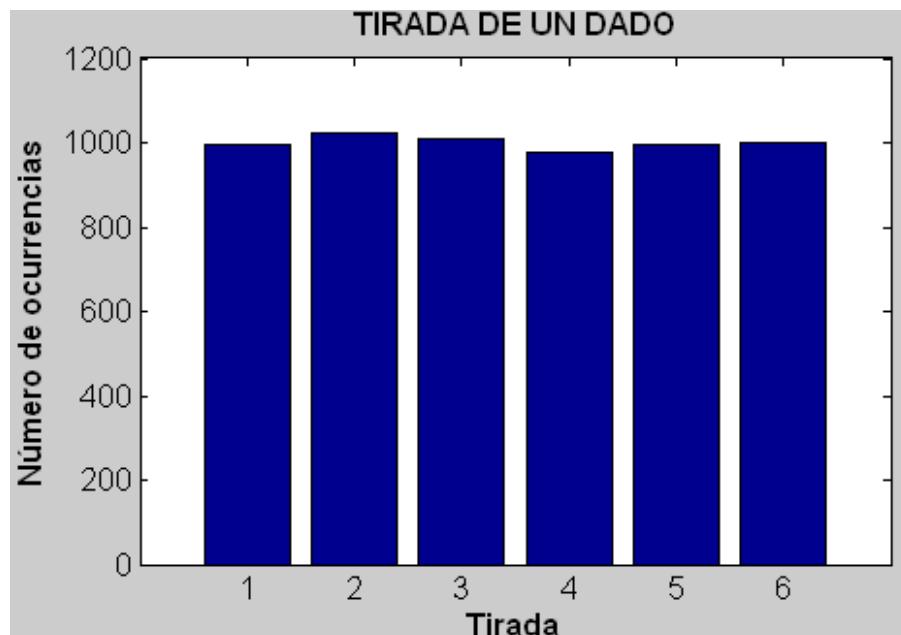
Función randi

- Generación de N números naturales aleatorios con distribución uniforme en el intervalo [Imin,Imax].

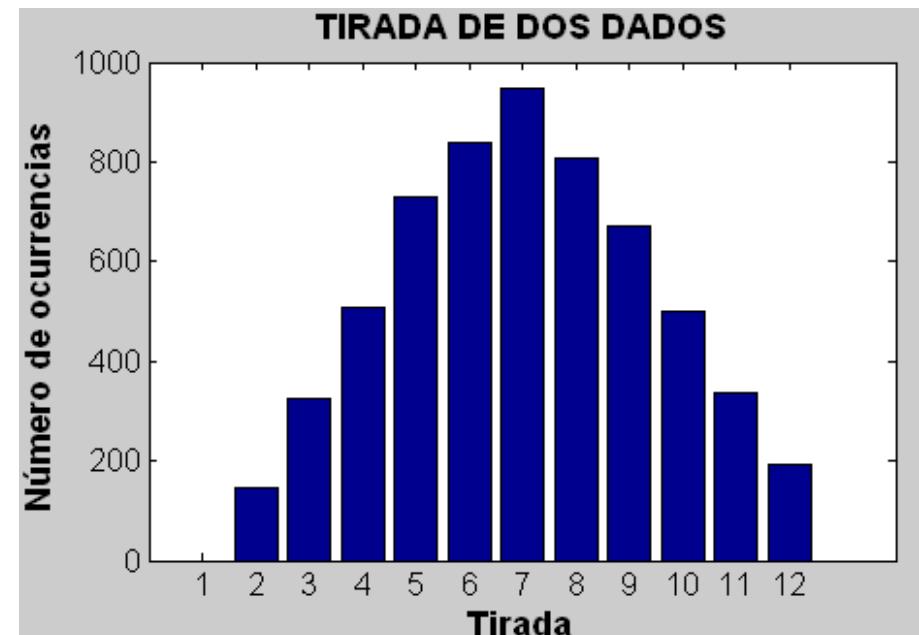
randi ([Imin , Imax] , M , N) → genera una matriz MxN



EJEMPLO_05.m

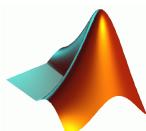


EJEMPLO_06.m



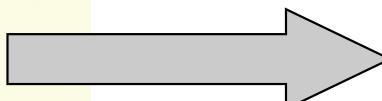
Función `randperm`

- Realiza una permutación aleatoria de números naturales.
`randperm(N)` → Permutación aleatoria de los naturales 1 a N.
- Útil en la aleatorización de los elementos de arrays.



EJEMPLO_07.m

```
% EJEMPLO 07: Aleatorización de listas  
% Uso de la función randperm.  
% Permutación aleatoria de índices  
clear; clc;  
CPs=(41005:41012)'  
r=randperm(length(CPs));  
CPs_rnd=CPs(r)
```



CPs =	CPs_rnd =
41005	41005
41006	41009
41007	41011
41008	41012
41009	41010
41010	41006
41011	41008
41012	41007



¿Aleatoriedad Determinista?

- Los ordenadores son deterministas → a las mismas entradas a un programa, las mismas salidas (salvo error).
 - ¿Cómo pueden entonces generar números aleatorios?
 - ¿Cómo de bien pueden hacerlo?
 - ¿Para qué sirve todo esto?
- Simulación de procesos aleatorios:
 - movimiento térmico
 - juegos de azar
 - desintegración radioactiva
 - solución estadística de ecuaciones
 - ...



Pseudo-Aleatoriedad

- Los ordenadores son deterministas → a las mismas entradas a un programa, las mismas salidas (salvo error).
 - ¿Cómo pueden entonces generar números aleatorios?

“Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin.”

John Von Neumann, 1951





Pseudo-Aleatoriedad

- Los ordenadores son deterministas → a las mismas entradas a un programa, las mismas salidas (salvo error).
 - ¿Cómo pueden entonces generar números aleatorios?
 - Estrictamente hablando, NO son capaces de generar secuencias aleatorias (sin correlación entre los números).
 - Si conocemos r_1, r_2, \dots, r_m , SIEMPRE es posible determinar r_{m+1} .
 - Realmente se generan **secuencias PSEUDO-aleatorias**.
 - ¿Cómo de bien pueden hacerlo?

Muy bien:

- Baja correlación
 - Periodo de repetitividad alto
- ... siempre que el generador sea bueno.



Generadores Congruentes Lineales (LCGs)

- Método más común para generar secuencias pseudo-aleatorias con **distribución uniforme**.



$$r_i \equiv (ar_{i-1} + c) \bmod M = \text{rem}\left(\frac{ar_{i-1} + c}{M}\right)$$

- Genera **números enteros en el intervalo [0,M-1]** a partir del resto del cociente (rem = remainder) de números grandes:
 - **r1 = semilla (seed)**, proporcionada por el usuario (o la máquina)
 - **M** = número muy grande (periodicidad máxima de la secuencia)
 - **a** = número grande; **c** = magia negra



Ejemplo de LCG

$$r_i = (ar_{i-1} + c) \bmod M, \text{ con } a = 57, c = 1, M = 256, r_1 = 10$$

- Secuencia pseudo-aleatoria en $[0, M-1]$:

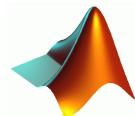
- $r_1 = 10$
- $r_2 = (57 \times 10 + 1) \bmod 256 = \text{rem}(571/256) = 59$
- $r_3 = (57 \times 59 + 1) \bmod 256 = \text{rem}(3364/256) = 36$
- ...
- $r_{257} = 10 \rightarrow$ Longitud de la secuencia (periodicidad $M = 256$)

- Secuencia pseudo-aleatoria en $[0, 1] \rightarrow r/M$

0.0391 0.2305 0.1406 0.0195 0.1172 0.6836 0.9688 0.2227 ...

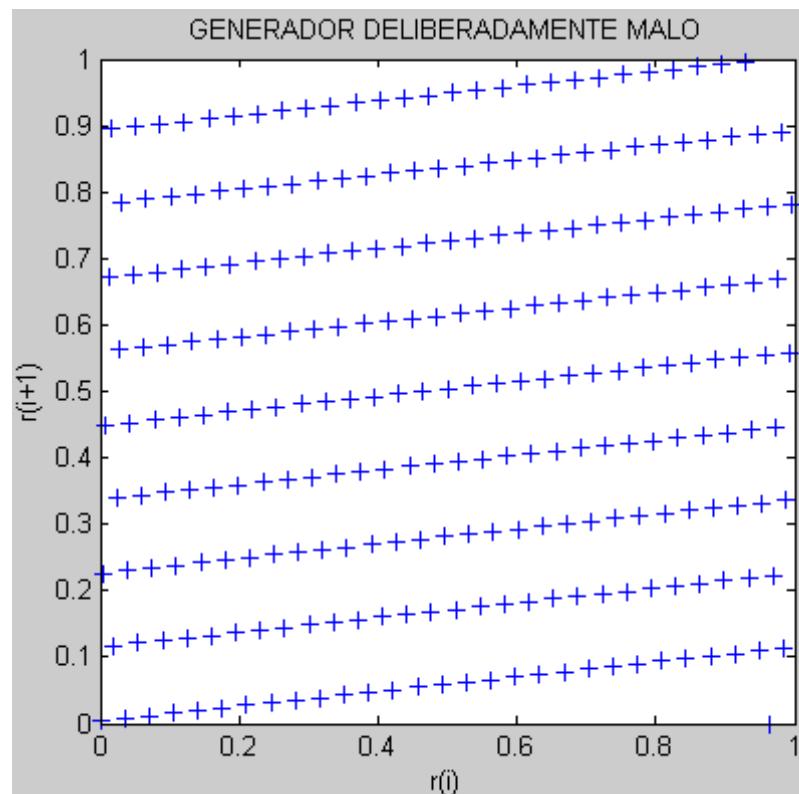
pero lógicamente con la misma periodicidad en la secuencia (M)

- Prueba simple para chequeo de secuencias → intentar reconocer visualmente patrones de correlación.

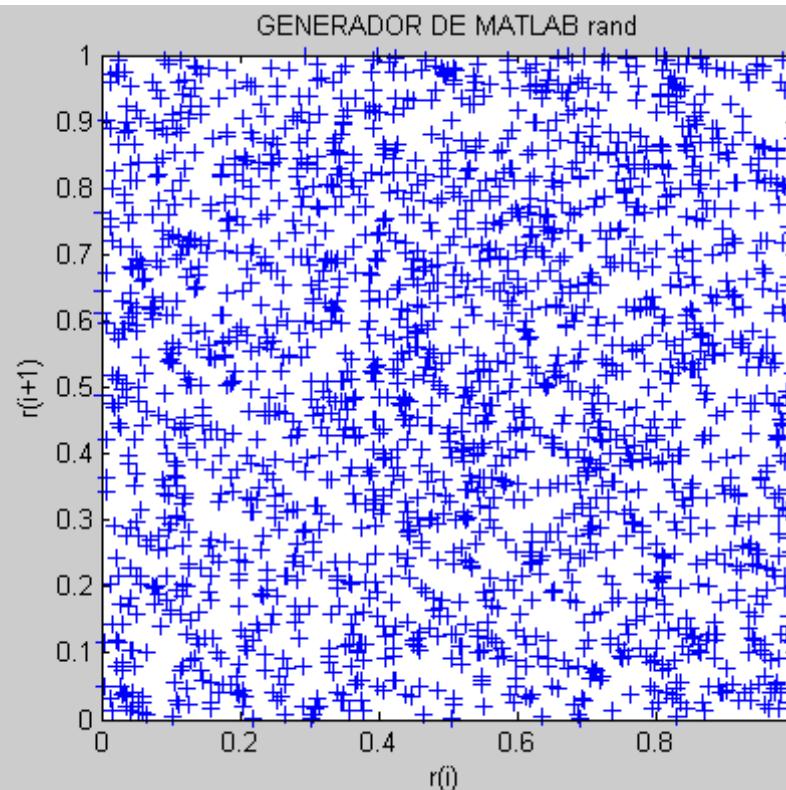


EJEMPLO_08.m (N = 2000)

LCG con $a = 57$, $c = 1$, $M = 256$
periodo 256



MATLAB rand → mt19937ar
Algoritmo Mersenne Twister, periodo $2^{19937}-1$



Notas
abajo

EJEMPLO_08b.m



Tests de Aleatoriedad y Uniformidad

1. Mirar el listado de números generados.
2. Dibujar la gráfica de scattering $r(i)$ frente i .
3. Evaluar el momento k -ésimo de la secuencia:

$$\langle x^k \rangle = \frac{1}{N} \sum_{i=1}^N x_i^k \cong \int_0^1 x^k P(x) dx + \Theta(1/\sqrt{N}) \cong \frac{1}{k+1}$$

Si se cumple la ecuación, la distribución es uniforme.

Si las desviaciones varían como $1/\sqrt{N}$, la distribución es además aleatoria.

4. Evaluar la correlación con vecinos próximos:

$$C(k) = \frac{1}{N} \sum_{i=1}^N x_i x_{i+k} \cong \int_0^1 dx \int_0^1 xy P(x, y) dy + \Theta(1/\sqrt{N}) \cong \frac{1}{4} \quad k = 1, 2, \dots$$

Si se cumple la ecuación, los números no están correlacionados.

Si las desviaciones varían como $1/\sqrt{N}$, la distribución es además aleatoria.



Tests de Aleatoriedad y Uniformidad (cont.)

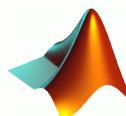
5. Dibujar la gráfica de scattering $r(2i+1)$ frente $r(2i)$.
6. Correr el cálculo o simulación con r_1, r_2, r_3, \dots y con $(1-r_1), (1-r_2), (1-r_3), \dots$
Los resultados no deben diferir más allá de la estadística del proceso.
7. Correr el cálculo o simulación con una secuencia de **números verdaderamente aleatorios tabulados** y comparar con el generador de números pseudo-aleatorios.
8. Realizar el test 3 para $k = 1, 3, 7$ y $N = 1e2, 1e4, 1e5$.
9. Realizar el test 4 a la serie levemente correlacionada $r_1, (1-r_1), r_2, (1-r_2), r_3, (1-r_3), \dots$ para $N = 1e2, 1e4, 1e5$.

...



Repetitividad en Secuencias Pseudo-Aleatorias

- A veces resulta útil poder correr un cálculo o simulación con **exactamente la misma secuencia pseudo-aleatoria**:
 - Para el **debugging** de scripts
 - Para chequear si el método computacional es correcto
- Para que una secuencia pseudo-aleatoria sea reproducible debemos usar siempre la **misma semilla**.
- Para cambiar la secuencia generada, basta usar una semilla distinta:
 - Cambio automático del estado tras una ejecución de la función en MATLAB
 - Uso del reloj del sistema como semilla



EJEMPLO_09.m



Monte Carlo (MC)

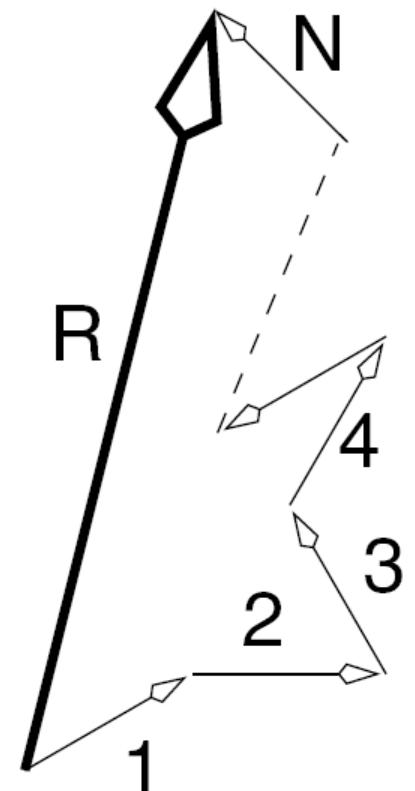
Familia de métodos numéricos para **solucionar problemas matemáticos o físicos mediante técnicas estadísticas** y que **conllevan el uso de un número alto de muestras aleatorias**

- **Simulación de procesos físicos aleatorios:**
 - Movimiento térmico
 - Desintegración radioactiva
- **Método numérico:**
 - Integración por valor medio
 - Integración por rechazo

Notas
abajo

Paseo Aleatorio (Random Walk)

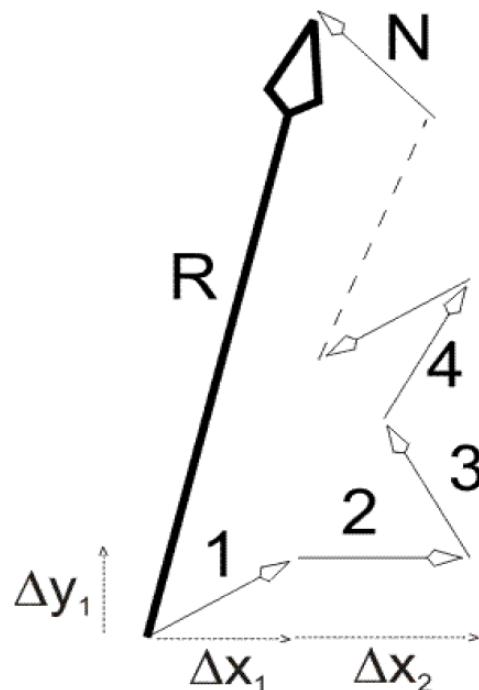
- Procesos físicos en los que una partícula parece moverse aleatoriamente:
 - Movimiento browniano
 - Fenómenos de transporte
- Problema: ¿Cuántas colisiones (en promedio) debe sufrir una partícula para recorrer una distancia radial R ?
- Modelo: Caminar N pasos de longitud r en direcciones aleatorias.





Paseo Aleatorio – Teoría

- ¿A qué distancia del origen llegamos después de N pasos?



$$\begin{aligned} R^2 &= (\Delta x_1 + \Delta x_2 + \dots + \Delta x_N)^2 + (\Delta y_1 + \Delta y_2 + \dots + \Delta y_N)^2 \\ &= \Delta x_1^2 + \dots + \Delta x_N^2 + 2\Delta x_1\Delta x_2 + 2\Delta x_1\Delta x_3 + 2\Delta x_2\Delta x_1 + \dots + (x \rightarrow y) \end{aligned}$$

- Si las direcciones son aleatorias, para N alto, los términos cruzados se anulan:

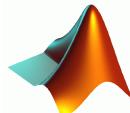
$$R^2 \cong \Delta x_1^2 + \dots + \Delta x_N^2 + \Delta y_1^2 + \dots + \Delta y_N^2 = N \langle r^2 \rangle$$

$$R_{rms} \cong \sqrt{Nr_{rms}}$$

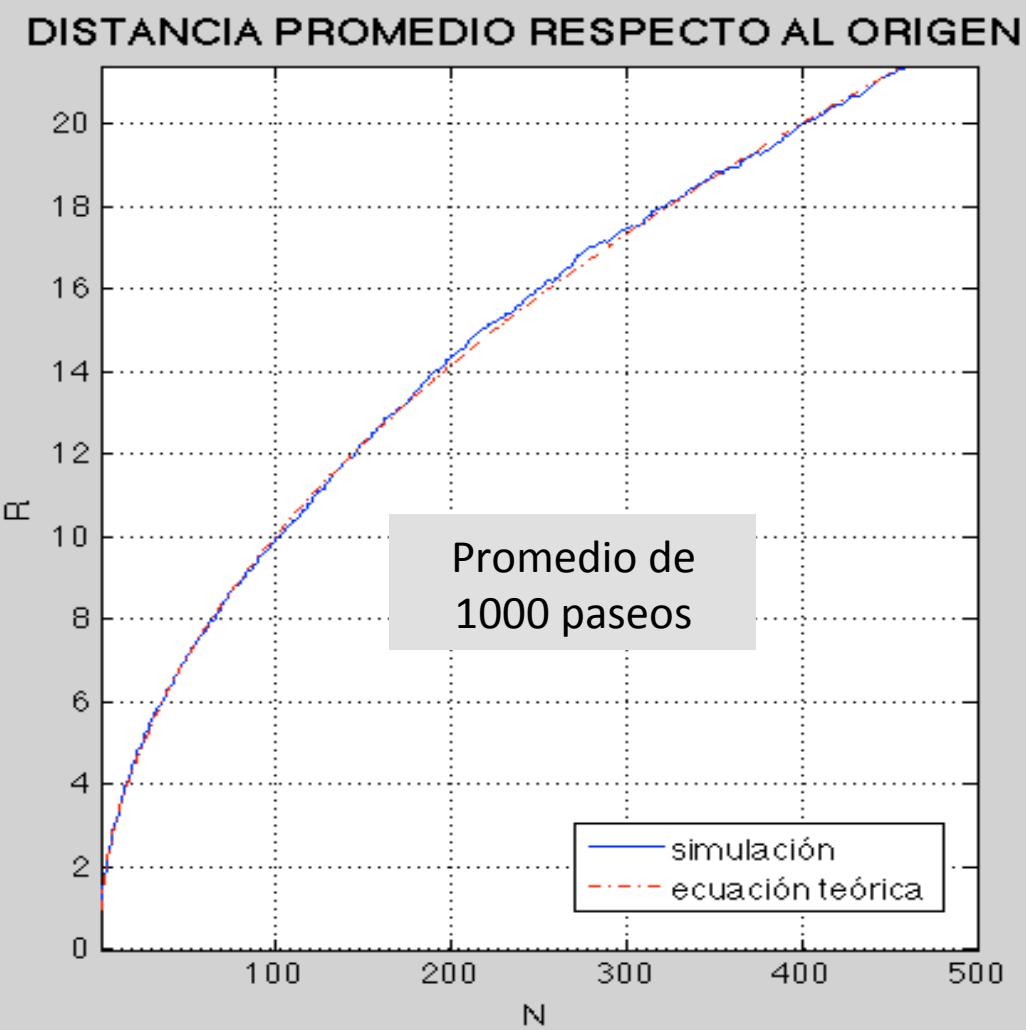
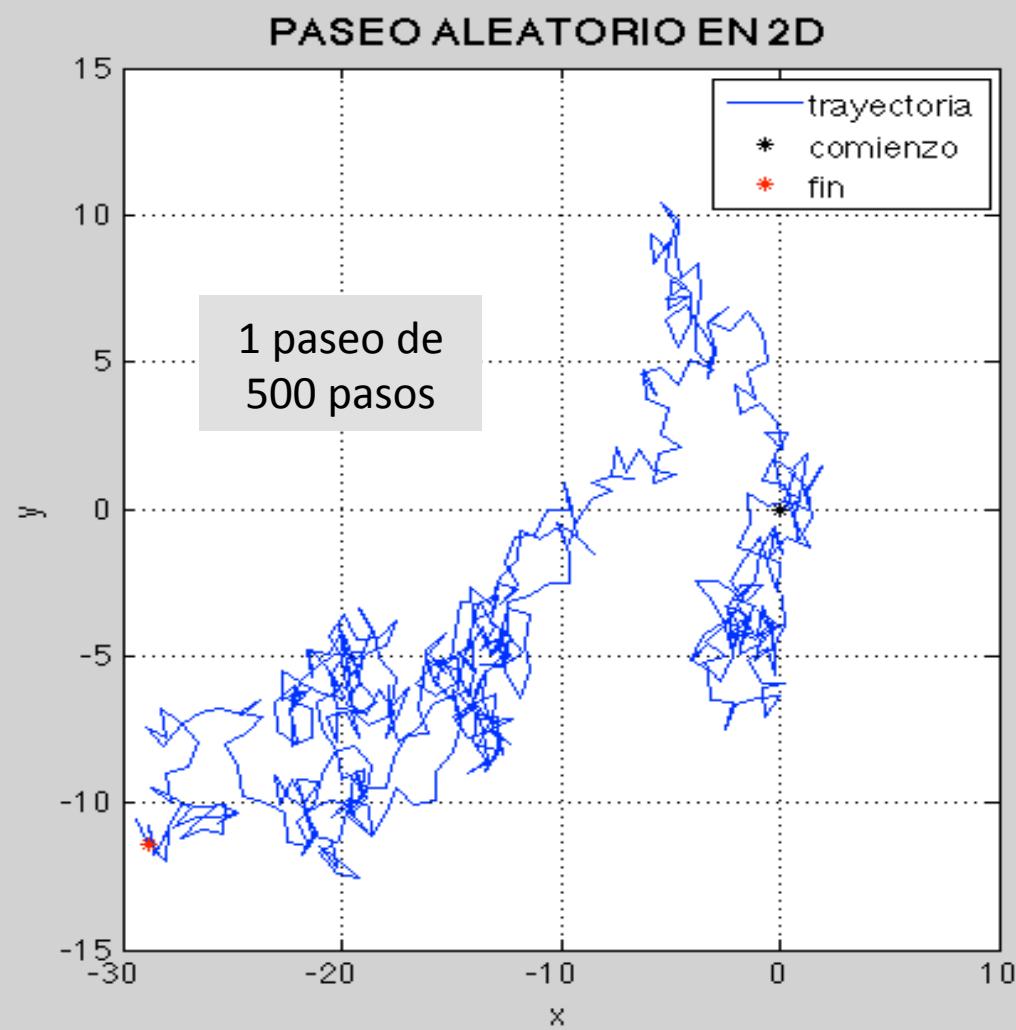
Si damos paseos de N pasos de longitud r en direcciones aleatorias, acabaremos los paseos (en promedio) a una distancia del origen proporcional a \sqrt{N} .



Paseo Aleatorio – Simulación



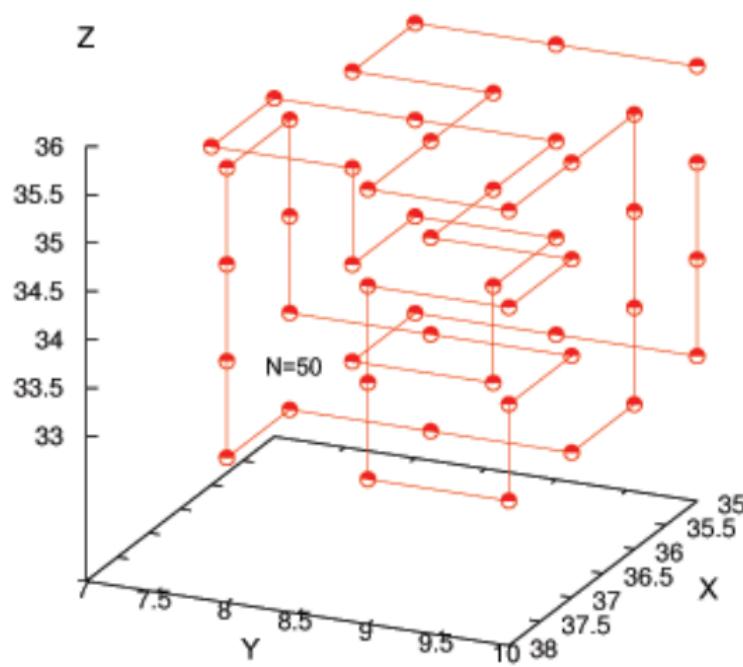
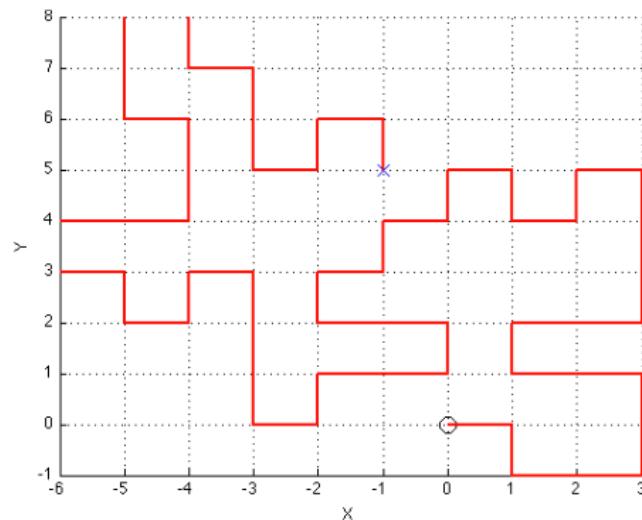
EJEMPLO_10.m





Self-Avoiding Walk (SAW)

- Paseo aleatorio en una retícula que no se cruza a sí mismo
- Útil en **crecimiento de polímeros, plegado de proteínas**, etc.
→ la forma geométrica resultante determina sus propiedades
 - Longitud del polímero
 - Distancia de extremo a extremo





Movimiento Browniano

- Sucesión de colisiones de pequeñas partículas (térmicas) con partículas mayores (brownianas)
- Útil en la **simulación** de procesos de **difusión, sedimentación**, etc.

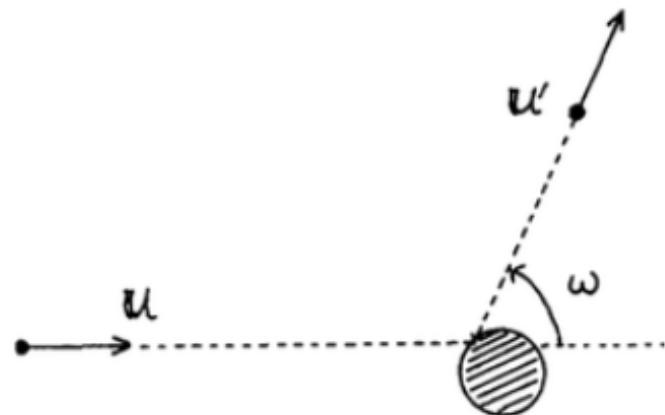


Ilustración de la colisión de una partícula térmica relativa a la browniana

$$\vec{u} = \vec{v}_t - \vec{V}_B \quad , \text{ con } |\vec{u}'| = |\vec{u}|$$
$$\vec{u}' = \vec{v}'_t - \vec{V}'_B \quad , \text{ con } |\vec{u}'| = |\vec{u}|$$

$$\vec{V}'_B = \vec{V}_B + \frac{m_t}{m_t + M_B} (\vec{u}' - \vec{u})$$



Notas
abajo

Desintegración Radioactiva

- Proceso natural en el que una partícula (átomo, núcleo), sin ningún estímulo externo, se desintegra en otras partículas.
- El instante en el que una partícula se desintegra es aleatorio. Es independiente de cuánto lleva existiendo la partícula o de qué le ocurre a otras a su alrededor.
 - La probabilidad $P(t)$ de desintegración por unidad de tiempo y por partícula es constante:

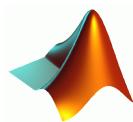
$$P(t) = \text{probab desinteg} / t / \text{partícula} = -\lambda$$

- Al ir disminuyendo el número de partículas N , lógicamente también lo hará el número de desintegraciones:

$$N(t), \Delta N / \Delta t \downarrow \text{con } t$$



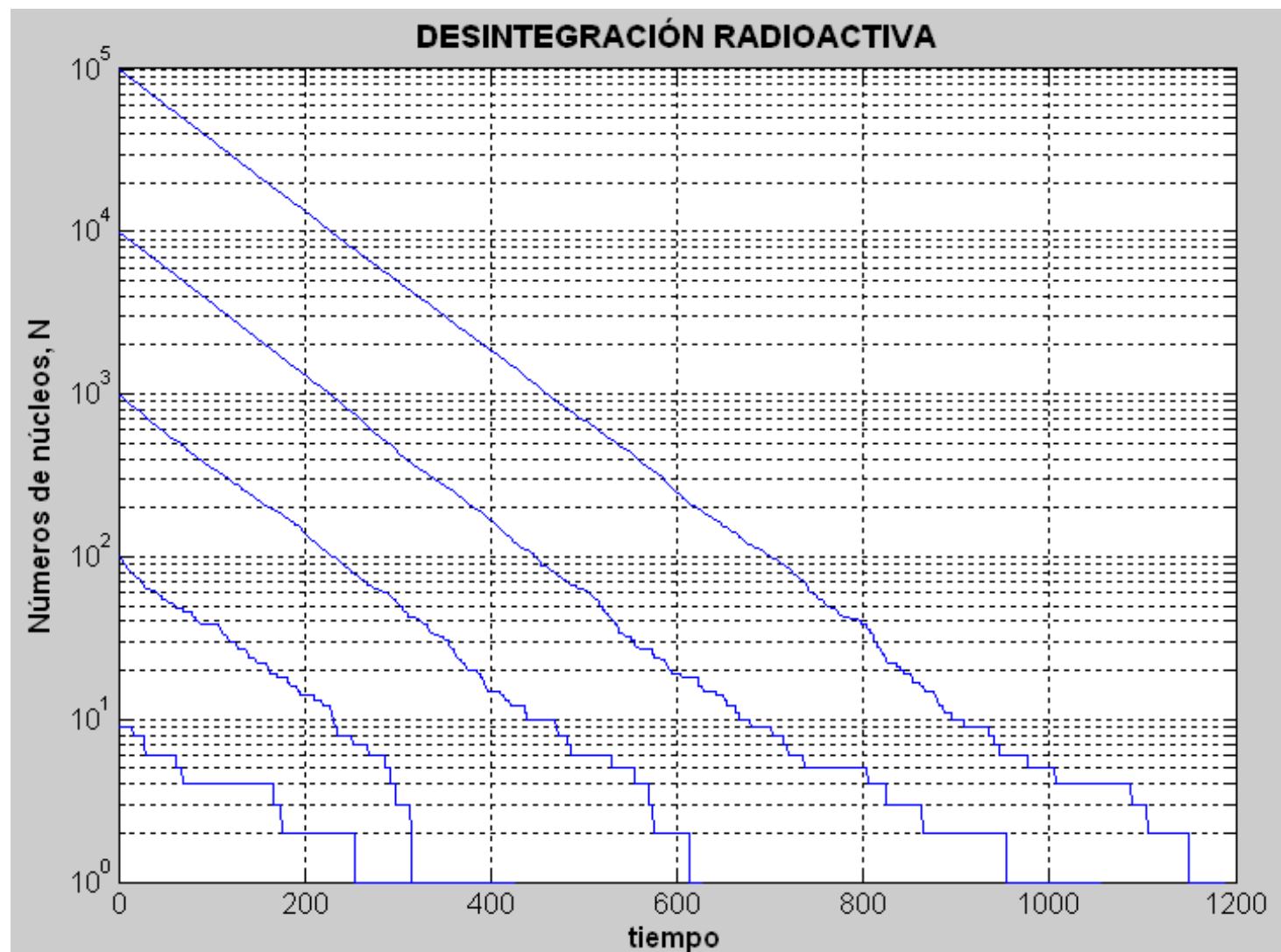
Desintegración Radioactiva – Simulación



EJEMPLO_11.m

La simulación del proceso demuestra:

- Ley de desintegración exponencial:
 $N(t) \propto e^{-\lambda t}$ si N alto
- Proceso estocástico si N bajo





Integración por Valor Medio

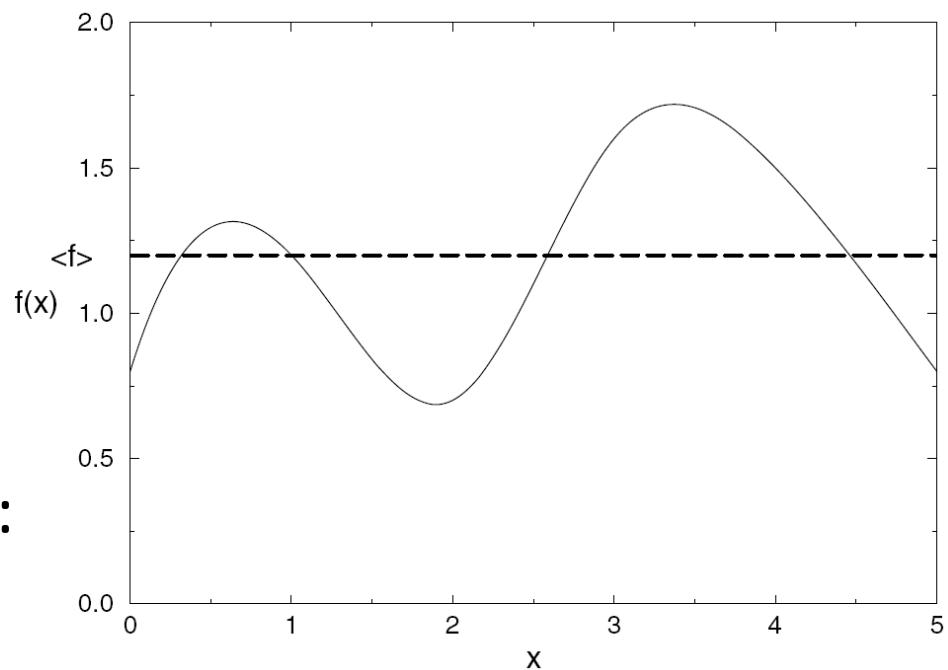
- Técnica estándar de integración por MC que está **basada en el Teorema del Valor Medio:**

$$I = \int_a^b f(x)dx = (b - a)\langle f \rangle$$

- Se usa una secuencia x_i de N números aleatorios en $[a,b]$ para **muestrear $\langle f \rangle$:**

$$\langle f \rangle \approx \frac{1}{N} \sum_{i=1}^N f(x_i)$$

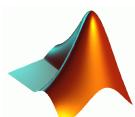
- El error en el valor de la integral disminuye como $1/\sqrt{N}$.





Integrales Multi-Dimensionales

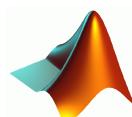
- La integración por valor medio **puede generalizarse** fácilmente a espacios multi-D sin aumentar la **complejidad** computacional:
$$I = \int_a^b dx \int_c^d dy \int_e^f f(x, y, z) dz = (b-a)(d-c)(f-e)\langle f \rangle$$
- Existen otros métodos de integración mejores para integrales simples y dobles, pero **cuando la dimensionalidad es alta las técnicas de MC son las mejores.**



EJEMPLO_12.m

Integración en 3D

$$I = \int_0^1 dx_1 \int_0^1 dx_2 \int_0^1 dx_3 (x_1 + x_2 + x_3)^2$$



EJEMPLO_13.m

Integración en 10D

$$I = \int_0^1 dx_1 \dots \int_0^1 dx_{10} (x_1 + \dots + x_{10})^2$$



Direct Sampling Method

- Problema formal:

Determinar la integral de una función $f(\mathbf{x})$ en un determinado volumen multi-dimensional V

$$I = \int_V f(\mathbf{x}) d\mathbf{x} \approx V\langle f \rangle \pm V \sqrt{\frac{\langle f^2 \rangle - \langle f \rangle^2}{N}}$$

- Método básico de estimación del valor medio:

Seleccionar N puntos aleatorios **uniformemente** distribuidos en ese volumen (x_1, x_2, \dots, x_N) y estimar la integral como

$$\langle f \rangle \equiv \frac{1}{N} \sum_{i=1}^N f(\mathbf{x}_i)$$

$$\langle f^2 \rangle \equiv \frac{1}{N} \sum_{i=1}^N f^2(\mathbf{x}_i)$$

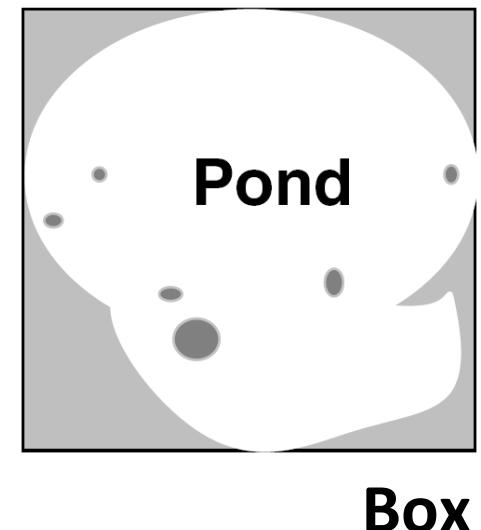
Notas
abajo



Método del Rechazo – Ejemplo

Tirar piedras en un estanque con forma irregular como método para determinar su área:

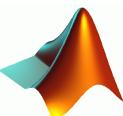
1. Señalice una caja que encierre completamente el estanque y quite todas las piedras que contenga.
2. Mida el área de esa caja (A_{box}).
3. Coja un buen montón de piedras y láncelas al aire en direcciones aleatorias.
4. Cuente el número de piedras que caen en el estanque (N_{pond}) y el número de piedras que caen al suelo dentro de la caja (N_{box}).
5. Asumiendo que tiró las piedras uniforme y aleatoriamente, N_{pond} debe ser proporcional a A_{pond} .



$$\frac{N_{pond}}{N_{pond} + N_{box}} = \frac{A_{pond}}{A_{box}} \quad \Rightarrow \quad A_{pond} = \frac{N_{pond}}{N_{pond} + N_{box}} A_{box}$$



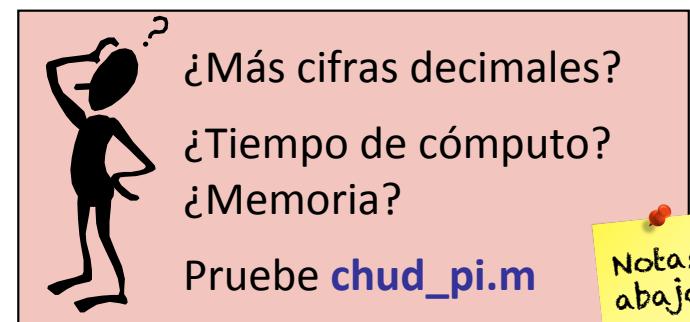
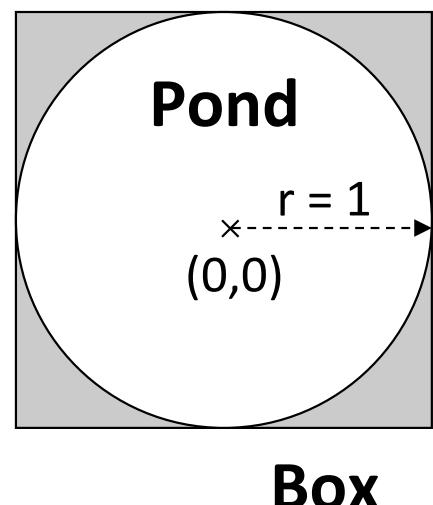
Método del Rechazo – Ejemplo



EJEMPLO_14.m

Determinar el número π utilizando la técnica de muestreo anterior.

1. Imagine un estanque circular encerrado en un cuadrado de lado 2.
2. Sabemos que $\int dA = \pi r^2 = \pi$
3. Generamos una secuencia de números aleatorios $\{r_i\}$ en $[-1, 1]$.
4. Asignamos $(x_i, y_i) = (r_{2i-1}, r_{2i})$ para $i = 1, \dots, N$.
5. Si $x_i^2 + y_i^2 < 1$, $N_{pond} = N_{pond} + 1$. En otro caso, $N_{box} = N_{box} + 1$.
6. Usar $A_{pond} = \frac{N_{pond}}{N_{pond} + N_{box}} A_{box} = \pi$
7. Aumente N hasta obtener π con 3 cifras decimales.





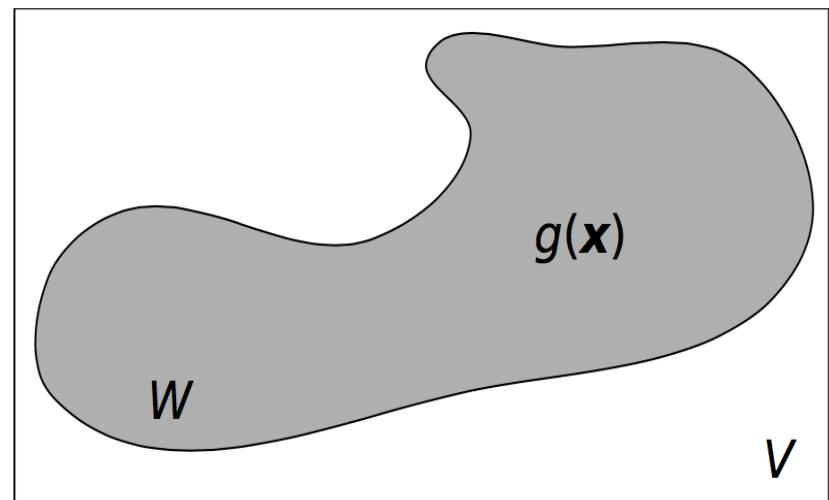
Método del Rechazo

- Determinar la integral de una función $g(\mathbf{x})$ en un determinado volumen multi-dimensional W sobre el cual no es fácil aplicar muestreo directo

$$I = \int_W g(\mathbf{x}) d\mathbf{x}$$



$$I = \int_W g(\mathbf{x}) d\mathbf{x} = \int_V f(\mathbf{x}) d\mathbf{x}$$



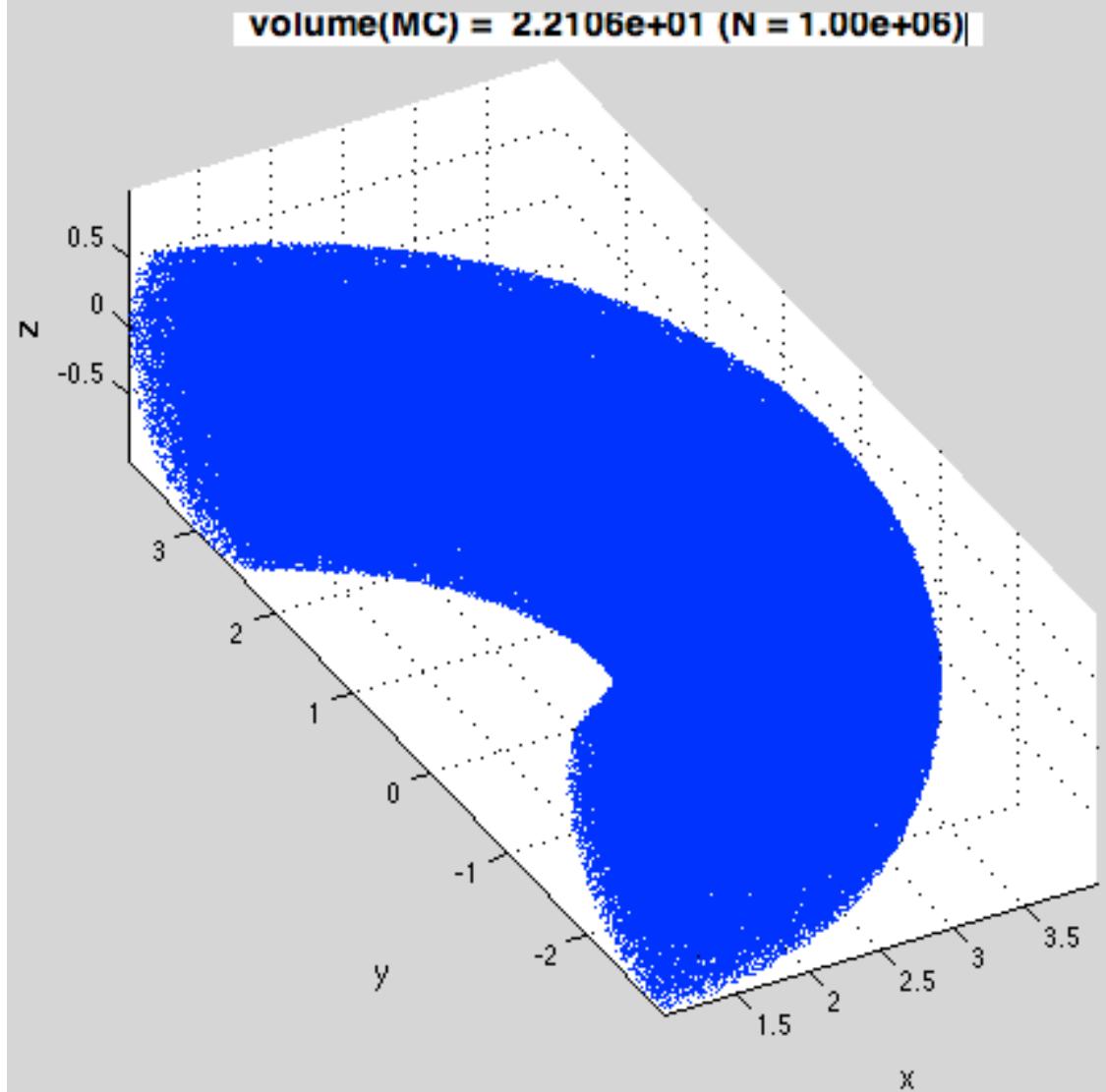
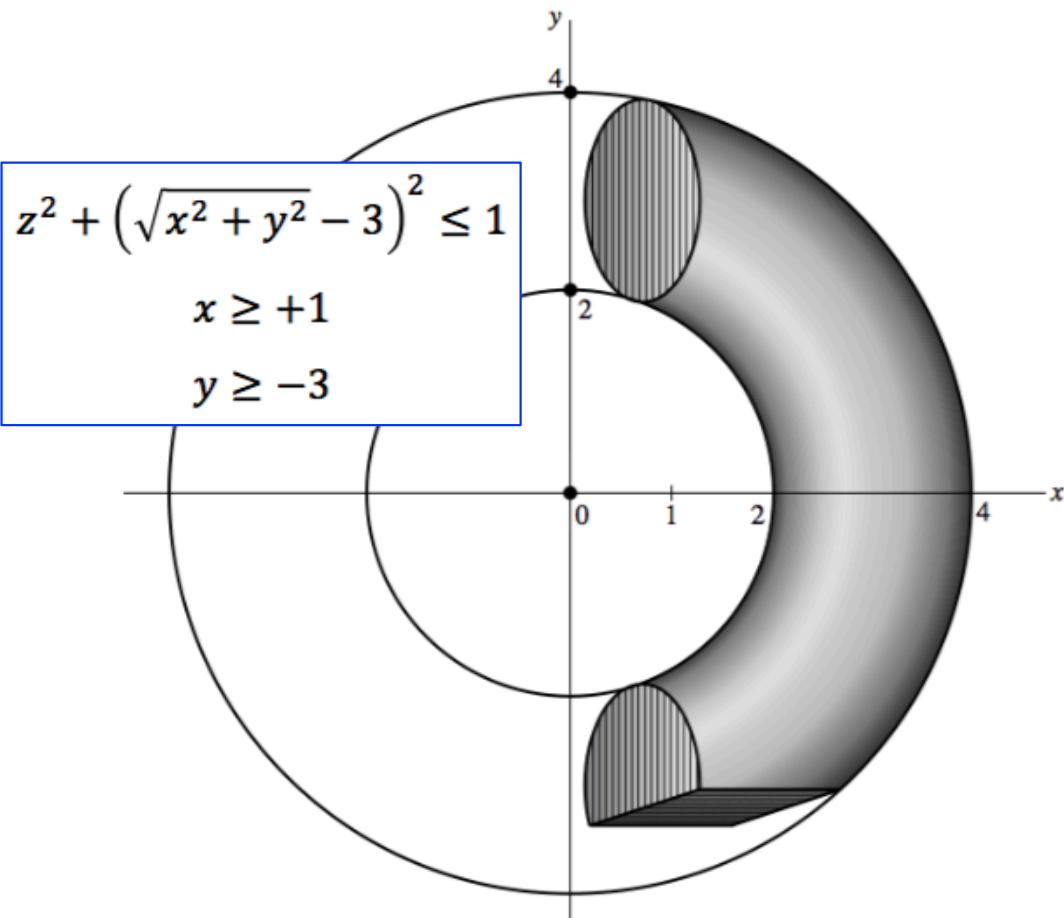
$$f(\mathbf{x}) = g(\mathbf{x}) \text{boolean}(\mathbf{x})$$

con $\text{boolean}(\mathbf{x}) = \begin{cases} 1, & \text{si } \mathbf{x} \text{ está dentro de } W \\ 0, & \text{si } \mathbf{x} \text{ está fuera de } W \text{ (y dentro de } V\text{)} \end{cases}$

Notas
abajo



Método del Rechazo – Ejemplo





Objetivos

- Saber utilizar las funciones `rand`, `randi` y `randn` de MATLAB para la generación de secuencias aleatorias.
- Comprender la naturaleza pseudo-aleatoria de cualquier secuencia generada de manera determinista por un ordenador.
- Ser capaz de aplicar tests simples para determinar la uniformidad y aleatoriedad de secuencias pseudo-aleatorias.
- Comprender la utilidad de los métodos de Monte Carlo en la simulación de procesos físicos estocásticos.
- Ser capaz de evaluar integrales multi-dimensionales mediante el muestreo del valor medio.