

## Tema 7: Métodos estocásticos

Ciertos problemas sólo pueden abordarse utilizando el azar y la estadística.

Fenómenos cuánticos  
(ejemplo: desintegración radioactiva)

Problemas de muchas partículas  
(ejemplo: movimiento browniano)

Problema modelo: difusión (gota de leche en una taza de café)

No interesan las trayectorias individuales, sino las magnitudes promedio.

Objetivo: establecer modelos estadísticos que tengan las mismas propiedades en promedio que el sistema real.

Camino aleatorio: trayectoria al azar en la que el "caminante" va cambiando aleatoriamente de dirección en cada paso.

# Tema 7: Métodos estocásticos

## 7.1 Generación de números aleatorios

Ordenador → generadores de números pseudo-aleatorios (PRNG)

C++: `rand( )`  
(entero entre 0 y `RAND_MAX`)

Python: `random( )`  
(real entre 0 y 1)

Requisito: los números generados deben estar uniformemente distribuidos.

Generador congruencial lineal:

$$x_{n+1} = (ax_n + c) \bmod(m)$$

Se necesita un valor inicial o semilla.

En Python es un valor relacionado con la hora del sistema.

En C++ está predeterminado ( $x_0=1$ ), pero se puede cambiar con

`srand(time(0))`

# Tema 7: Métodos estocásticos

## 7.1 Generación de números aleatorios

En realidad, Python no usa el generador congruencial lineal, sino otro generador denominado "Mersenne Twister", o Tornado de Mersenne.

Está basado en los números primos de Mersenne:  $2^n - 1$

El más utilizado usa  $2^{19937} - 1$ . En C++ se puede utilizar con la función "mt19937".

Para usar este generador de números aleatorios en Python tenemos que importar el paquete "random":

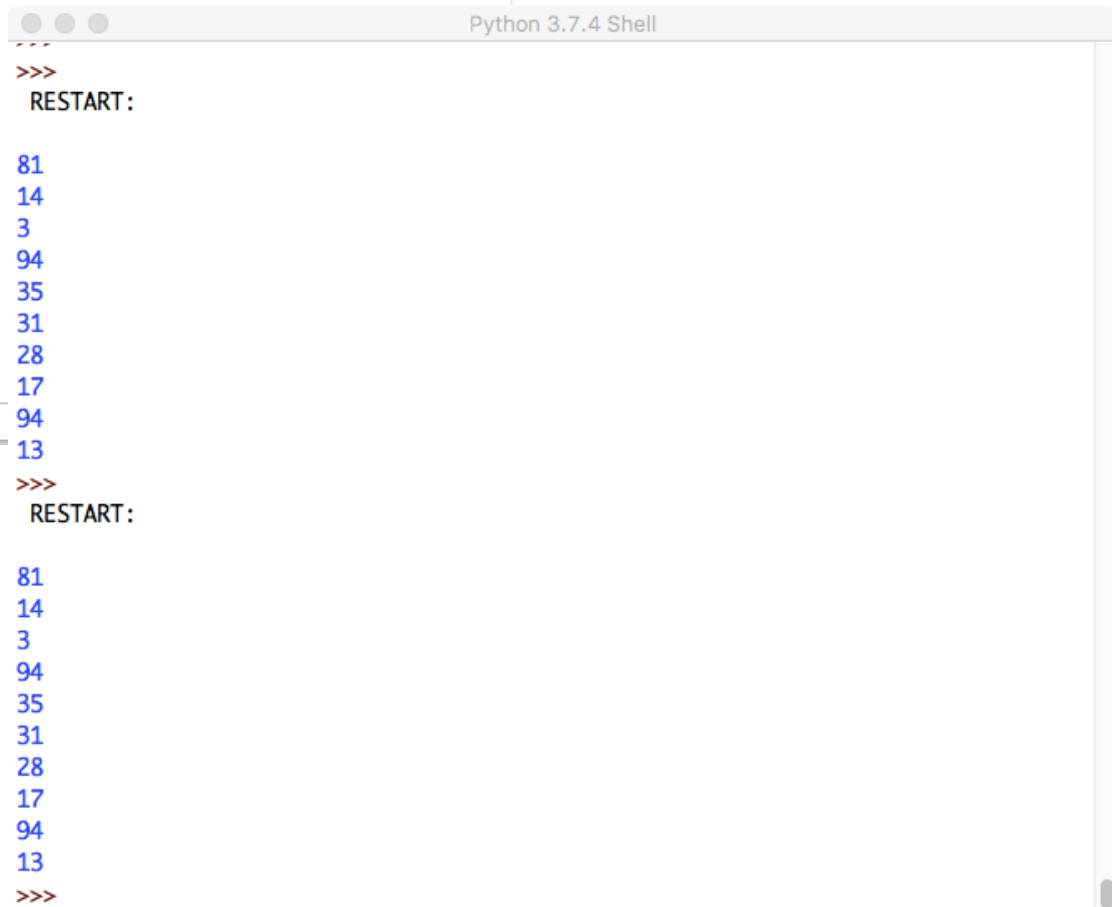
```
from random import *
a=1
lista=["a","e","i","o","u"]
while a!=0:
    print("Número real entre 0 y 1, el 1 no cuenta:", random() )
    print("Número entero entre 0 y 99:", randrange(100) )
    print("Número entero entre 50 y 99:", randrange(50,100) )
    print("Número entero entre 50 y 99, múltiplo de 5:", randrange(50,100,5) )
    print("Número real entre 50 y 100, el 100 no cuenta:", uniform(50,100) )
    print("Elemento al azar de la lista",lista,":", choice(lista) )
    print("Número entero entre 50 y 100, el 100 sí que entra:", randint(50,100) )
    a=int(input("\nPara parar introduce 0, para seguir, cualquier otro número:"))
```

Para fijar la semilla en Python a un valor fijo: `seed( )`. Útil para depurar.

# Tema 7: Métodos estocásticos

## 7.1 Generación de números aleatorios

```
from random import *  
  
seed(42)  
for i in range(10):  
    print(randrange(100))
```



```
Python 3.7.4 Shell  
>>>  
RESTART:  
  
81  
14  
3  
94  
35  
31  
28  
17  
94  
13  
>>>  
RESTART:  
  
81  
14  
3  
94  
35  
31  
28  
17  
94  
13  
>>>
```

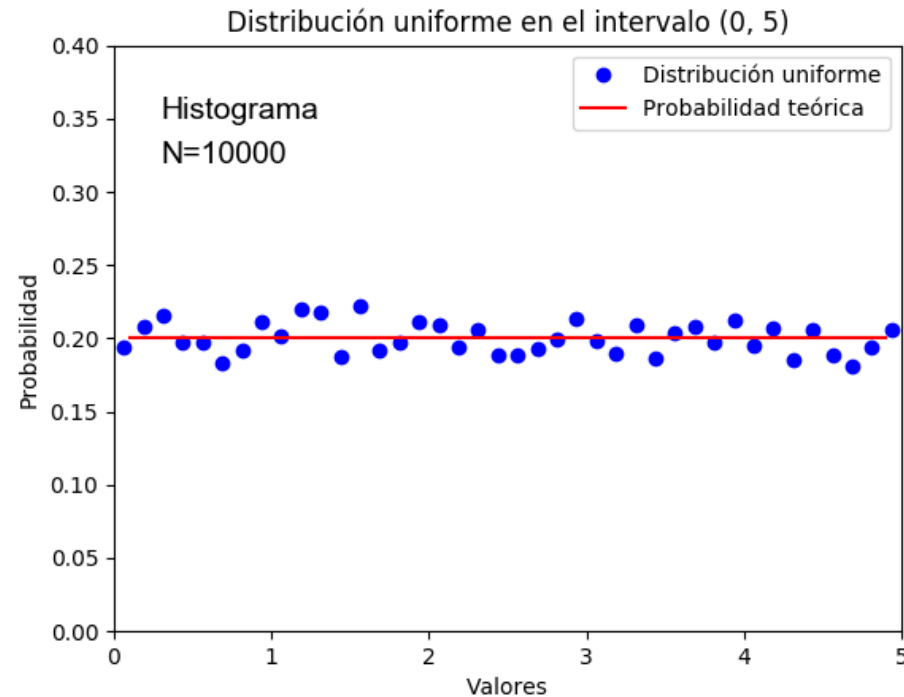
Ln: 76180 Col: 0

# Tema 7: Métodos estocásticos

## 7.1 Generación de números aleatorios

### **Distribuciones de números aleatorios no uniformes**

No siempre se busca generar una secuencia de números uniforme.

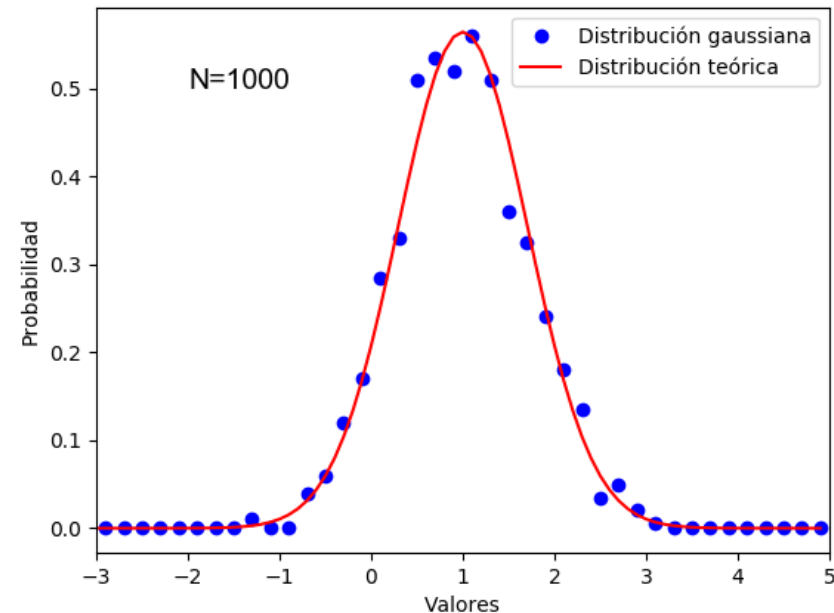
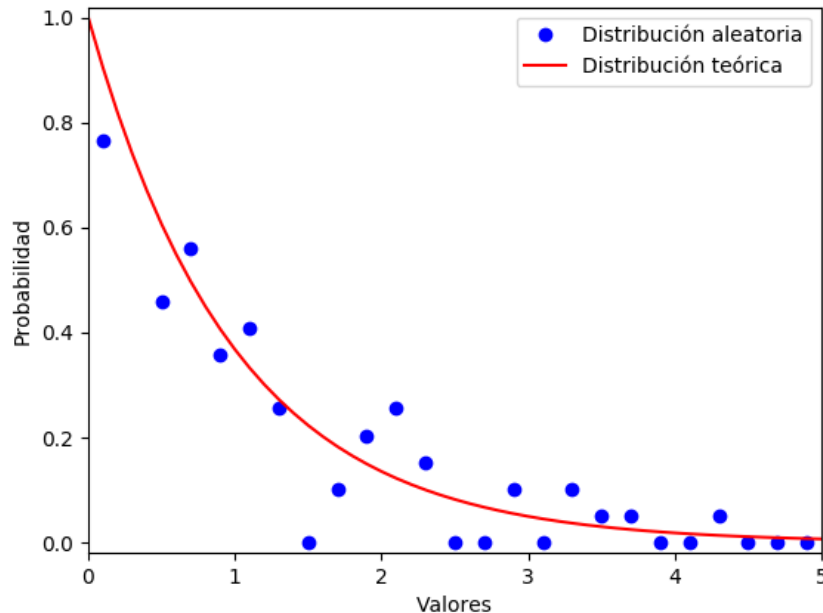


# Tema 7: Métodos estocásticos

## 7.1 Generación de números aleatorios

### **Distribuciones de números aleatorios no uniformes**

No siempre se busca generar una secuencia de números uniforme. Por ejemplo:



El Método de la Transformación consiste en relacionar la distribución que queremos usar con otra distribución conocida, como puede ser la uniforme.

El Método del Rechazo es más general y consiste en generar una secuencia uniforme y rechazar valores de acuerdo a la probabilidad que nos da la distribución que queremos producir.

# Tema 7: Métodos estocásticos

## 7.1 Generación de números aleatorios

### Representación con histogramas en Python

```
num_clases=25                # Número de recipientes para el histograma
val_max=5.0                  # Máximo valor que vamos a considerar

# Usamos la función histogram de numpy. Devuelve un array con las cuentas de cada clase y
# otro con los límites entre clases. La dimensión de éste último es una unidad mayor que
# número de clases

cuentas,lims_clases=np.histogram(valores,bins=num_clases,range=(0,val_max),density=True)

# Creamos un array (clases) con valores centrados en cada clase

clases = lims_clases[:-1].copy()    # Copia de lims_clases con el último elemento borrado
clases = clases+(1/2)*val_max/num_clases    # Sumamos la mitad del tamaño de cada clase

# Creamos el gráfico:

plot(clases,cuentas,"ob", label="Distribución aleatoria")

# Creamos una curva con la distribución teórica y la añadimos al plot

x_teo=np.arange(0,5.1,0.1)
y_teo=np.exp(-x_teo)
plot(x_teo,y_teo,"-r", label="Distribución teórica")

legend()    # Para incluir las leyendas que figuran en los "labels"
xlabel("Valores")
ylabel("Probabilidad")
xlim(0,5)
ylim(-0.02,1.02)
show()
```

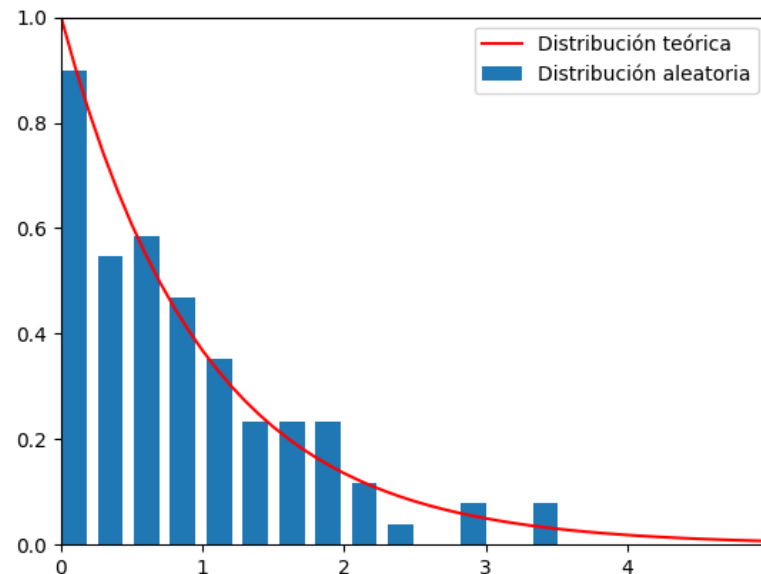
# Tema 7: Métodos estocásticos

## 7.1 Generación de números aleatorios

### Representación con histogramas en Python

También podemos usar la función "hist" incluida en el paquete matplotlib.pyplot, aunque de esta manera se obtiene un gráfico de barras.

```
num_clases=25                                # Número de recipientes para el histograma
val_max=5.0                                  # Máximo valor que vamos a considerar
hist(valores,num_clases,density=True,width=0.9*val_max/num_clases, label="Distribución
aleatoria")
plot(x_teo,y_teo,"-r", label="Distribución teórica")
legend()
xlim(0,5)
ylim(0,1)
show()
```





# Tema 7: Métodos estocásticos

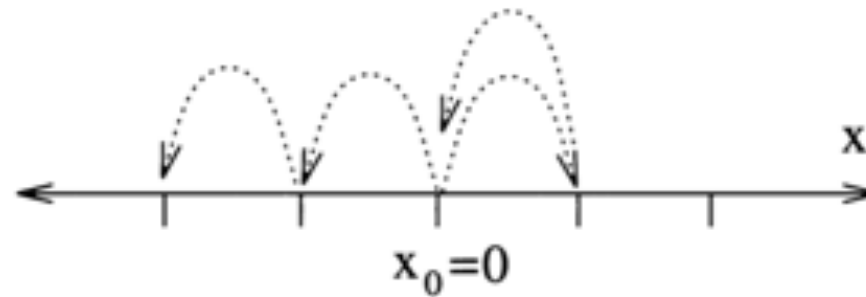
## 7.1 Generación de números aleatorios

Para algunas funciones comunes, Python ofrece la posibilidad de generar número aleatorios según las correspondientes distribuciones. En los ejercicios anteriores serían `expovariate(1)` y `gauss(1,1)`.

# Tema 7: Métodos estocásticos

## 7.2 Caminos aleatorios

Camino aleatorio unidimensional

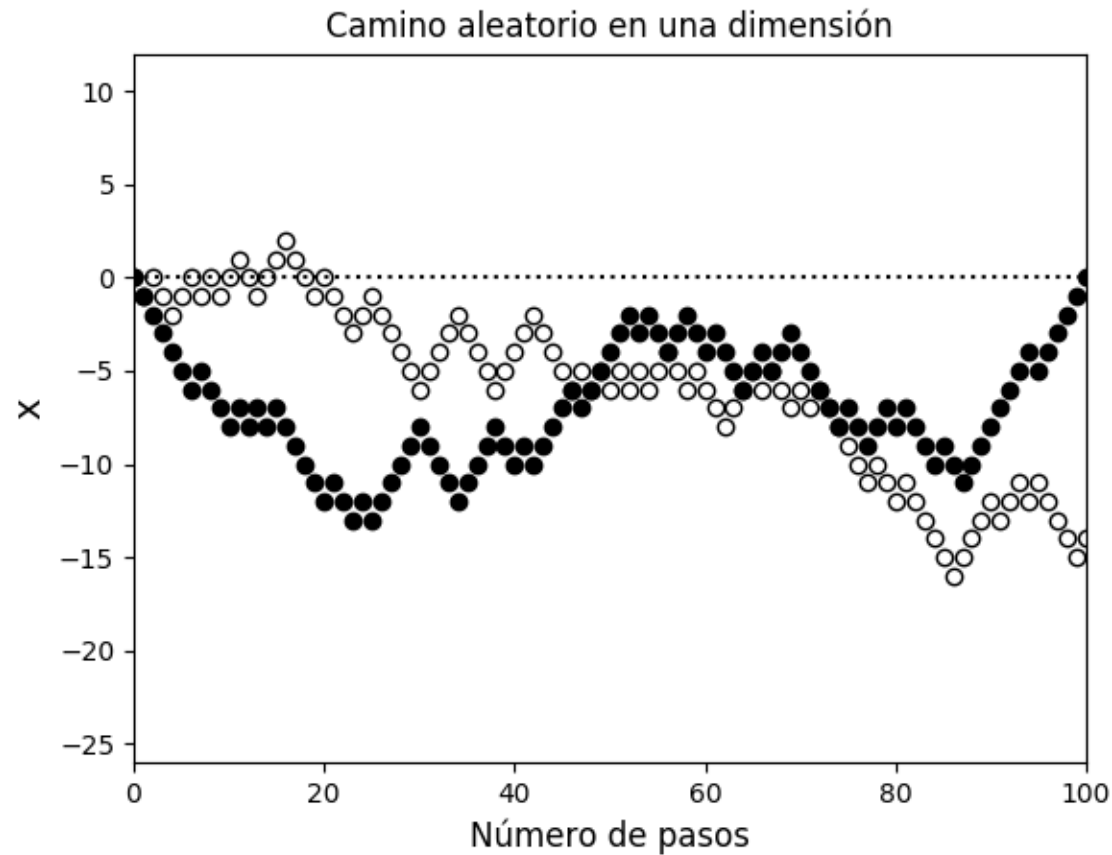


Intervalo entre pasos  $\sim$  constante: escala temporal  $\propto n^\circ$  pasos

```
from random import *
N_pasos=100
x=0
camino=[0.0]
for j in range(N_pasos):
    paso=random()
    if paso < 0.5:
        x=x+1
    else:
        x=x-1
    camino.append(x)
```

# Tema 7: Métodos estocásticos

## 7.2 Caminos aleatorios



Para un número suficientemente grande de caminos,  $\langle x_n \rangle \rightarrow 0$

# Tema 7: Métodos estocásticos

## 7.2 Caminos aleatorios

Algoritmo para calcular el promedio de  $x^2$  para varios caminos:

- 1) Establecer  $N_{\text{caminos}}$  y  $N_{\text{pasos}}$  de cada camino.
- 2) Crear un array con  $N_{\text{pasos}}+1$  elementos para guardar los valores de  $\langle x^2 \rangle$ :  $x2$
- 3) Crear un bucle en  $i$  sobre el  $n^\circ$  de caminos. En cada ejecución crear un camino inicializando  $x=0$  y generando el camino con  $N_{\text{pasos}}$  (bucle en  $j$ )

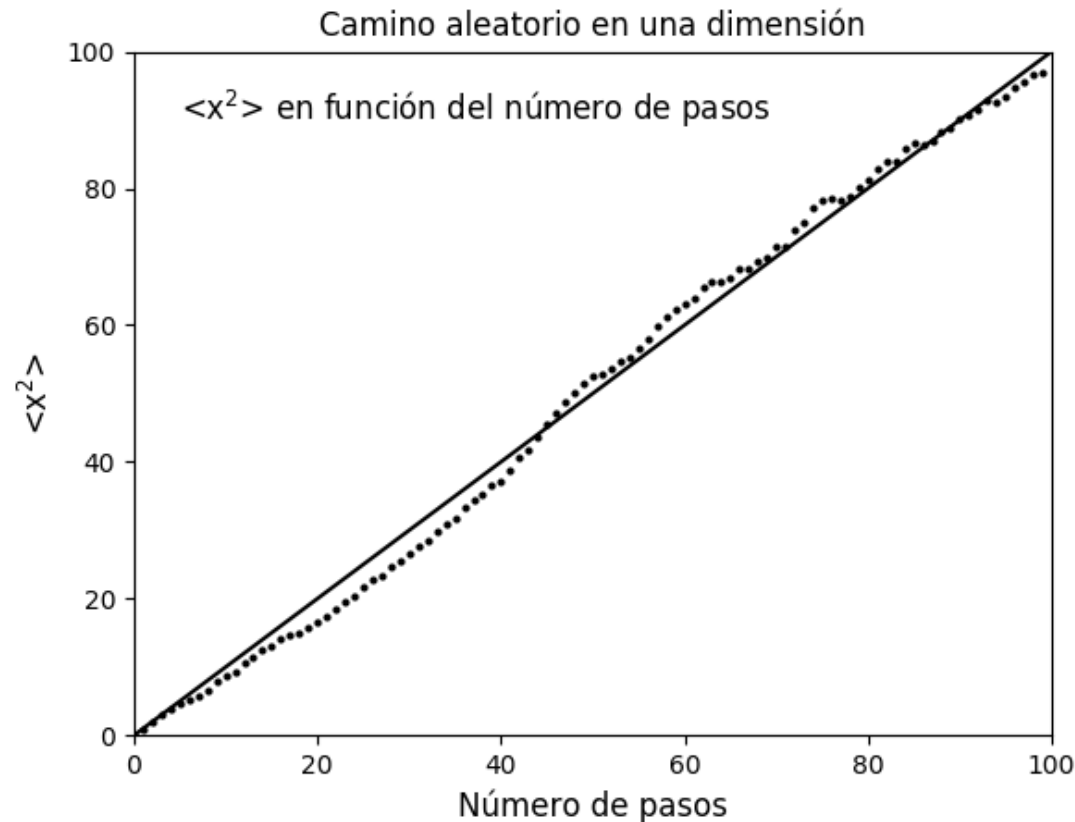
4) En cada paso del bucle en  $j$  de cada camino ir acumulando el correspondiente valor de  $x2[j] = x2[j] + x^2$

5) Una vez recorridos todos los caminos, normalizar el valor de  $x2$

```
sub calculate(x2ave(),n_walks,n_steps)
  plot_flag = 1
  for i = 1 to n_walks
    x = 0                                ! current location of the walker
    for j = 1 to n_steps
      if rnd < 0.5 then
        x = x + 1
      else
        x = x - 1                        ! just use an else statement
        ! DO NOT generate another new value using rnd
      end if
      x2ave(j) = x2ave(j) + x^2
    next j
  next i
  for i = 1 to n_steps                    ! normalize x2ave when finished
    x2ave(i) = x2ave(i) / n_walks
  next i
end sub
```

# Tema 7: Métodos estocásticos

## 7.2 Caminos aleatorios



Camino aleatorio:  $\langle x^2 \rangle = D \cdot t \Rightarrow \sqrt{\langle x^2 \rangle} \sim t^{1/2}$       Partícula libre:  $x = v \cdot t$

Si duplicamos el diámetro de la taza:  $d_1 = D \cdot t_1^{1/2}$  ;  $d_2 = D \cdot t_2^{1/2} = 2d_1 = 2D \cdot t_1^{1/2}$

$$t_2 = 4t_1$$

# Tema 7: Métodos estocásticos

## 7.2 Caminos aleatorios

$$x_n = \sum_{i=1}^n s_i ; \quad s_i = \pm 1$$

$$x_n^2 = \left( \sum_{i=1}^n s_i \right) \cdot \left( \sum_{j=1}^n s_j \right) = \sum_{i=1}^n \left( \sum_{j=1}^n s_i s_j \right)$$

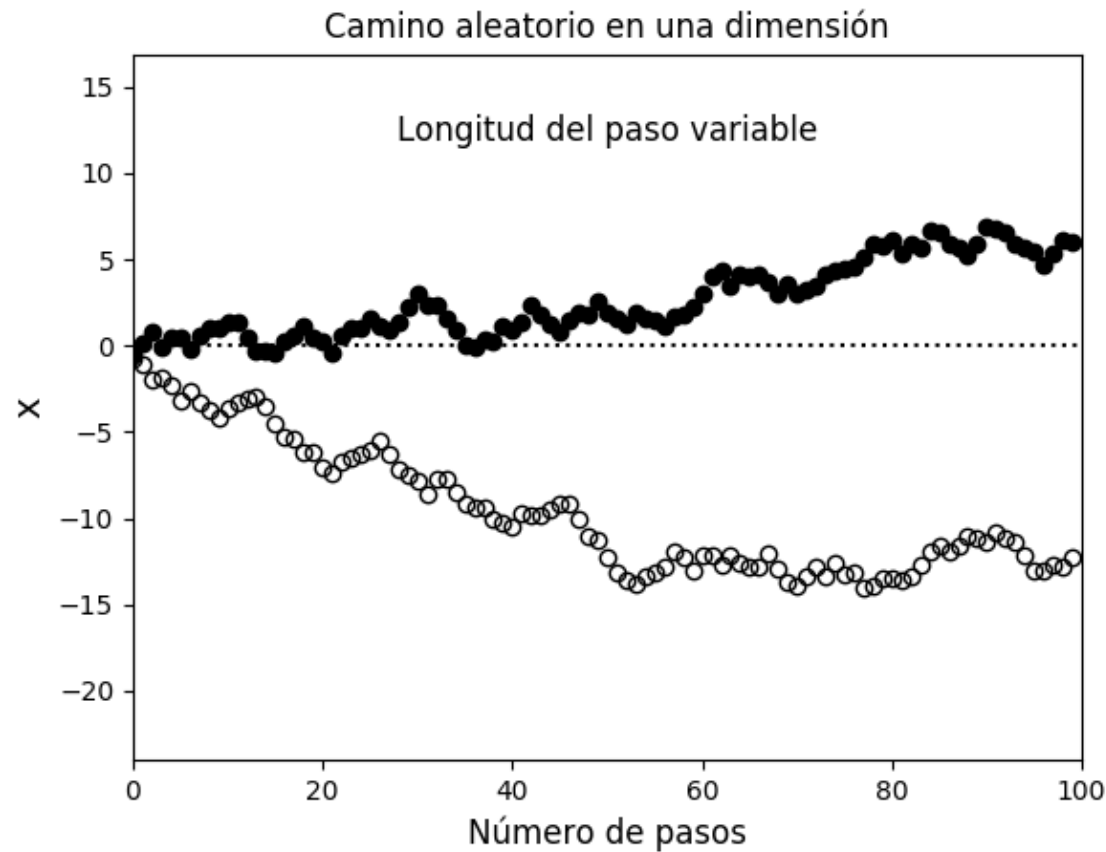
Para  $n$  grande, e  $i \neq j$ , los términos  $s_i s_j$  se cancelan.

$$\langle x_n^2 \rangle = \sum_{i=1}^n s_i^2 = n \Rightarrow \langle x_n^2 \rangle = t \Rightarrow D = 1$$

# Tema 7: Métodos estocásticos

## 7.2 Caminos aleatorios

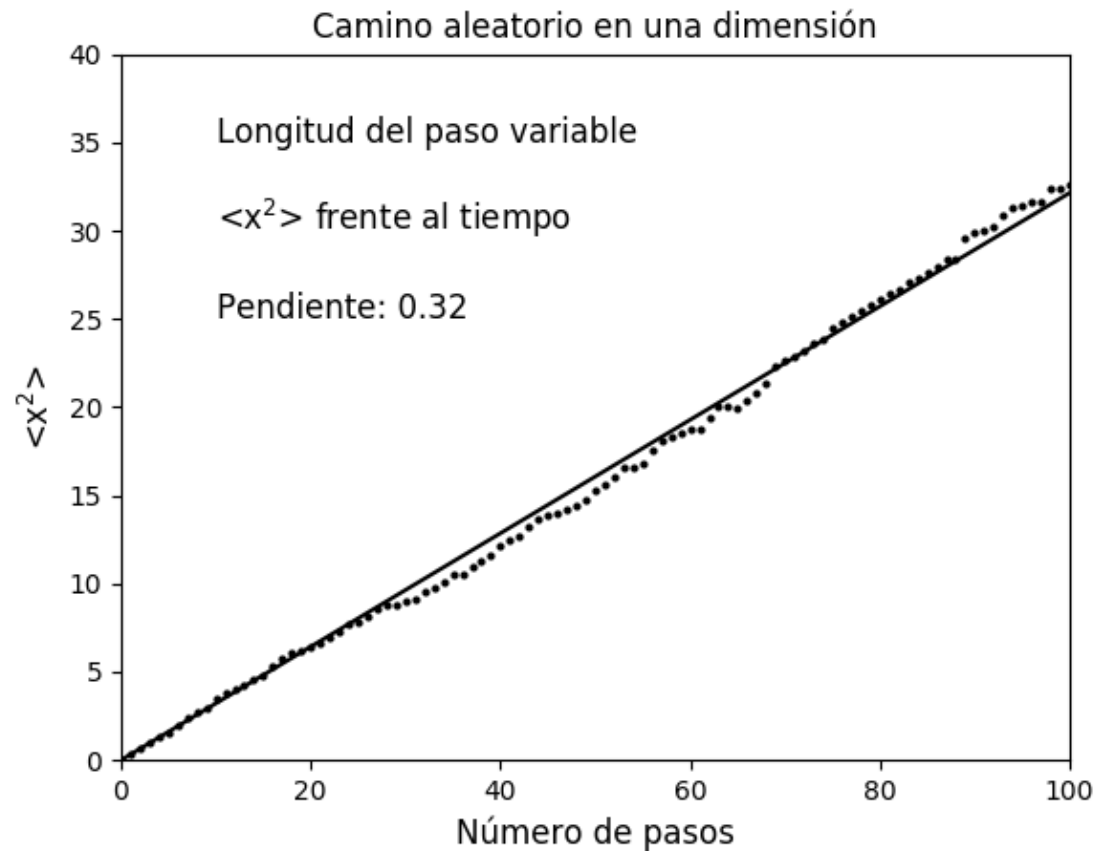
Camino aleatorio unidimensional con longitud de paso variable



# Tema 7: Métodos estocásticos

## 7.2 Caminos aleatorios

Camino aleatorio unidimensional con longitud de paso variable



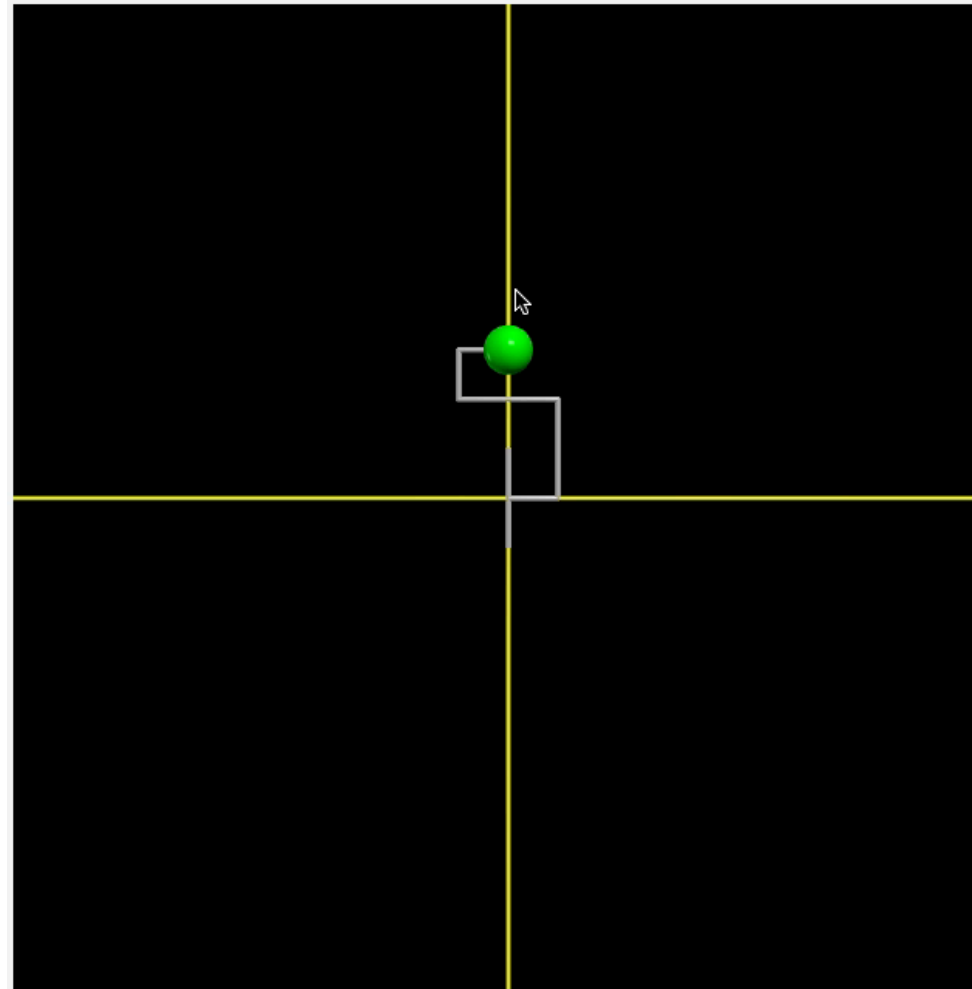
Comportamiento difusivo con  $D < 1$



# Tema 7: Métodos estocásticos

## 7.2 Caminos aleatorios

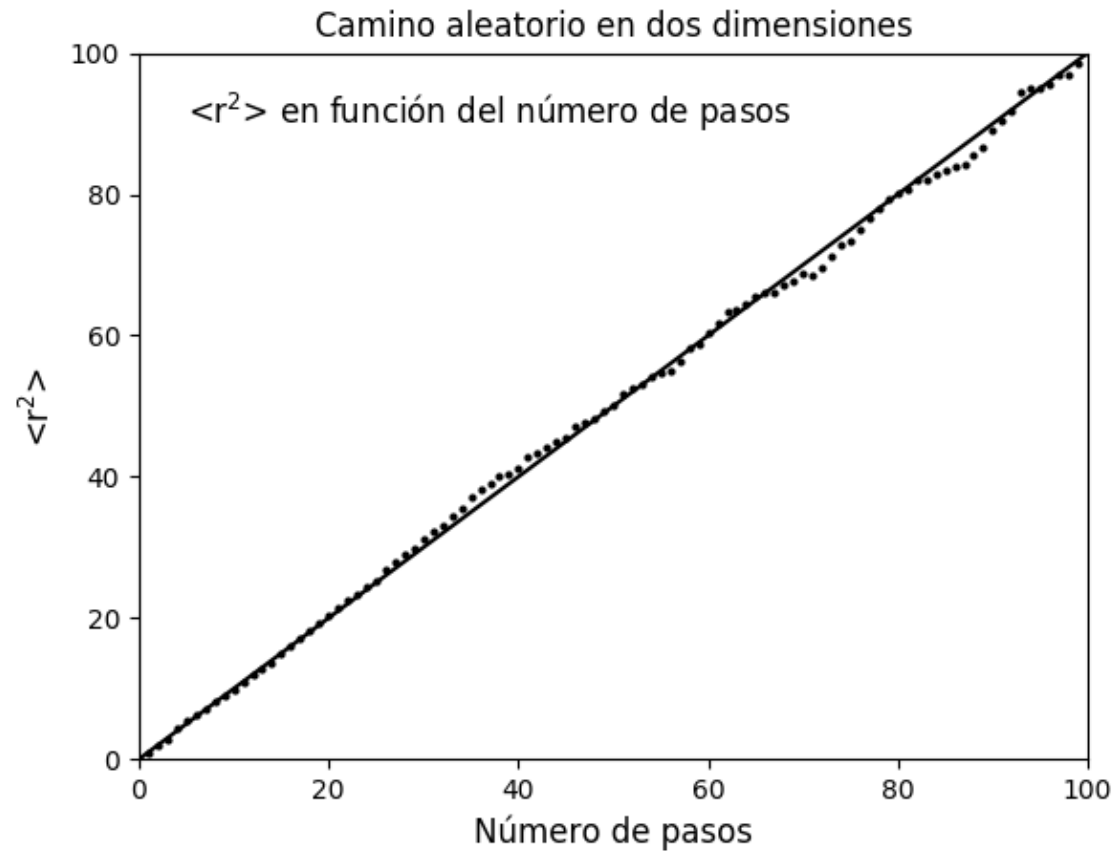
Camino aleatorio en 2 dimensiones



# Tema 7: Métodos estocásticos

## 7.2 Caminos aleatorios

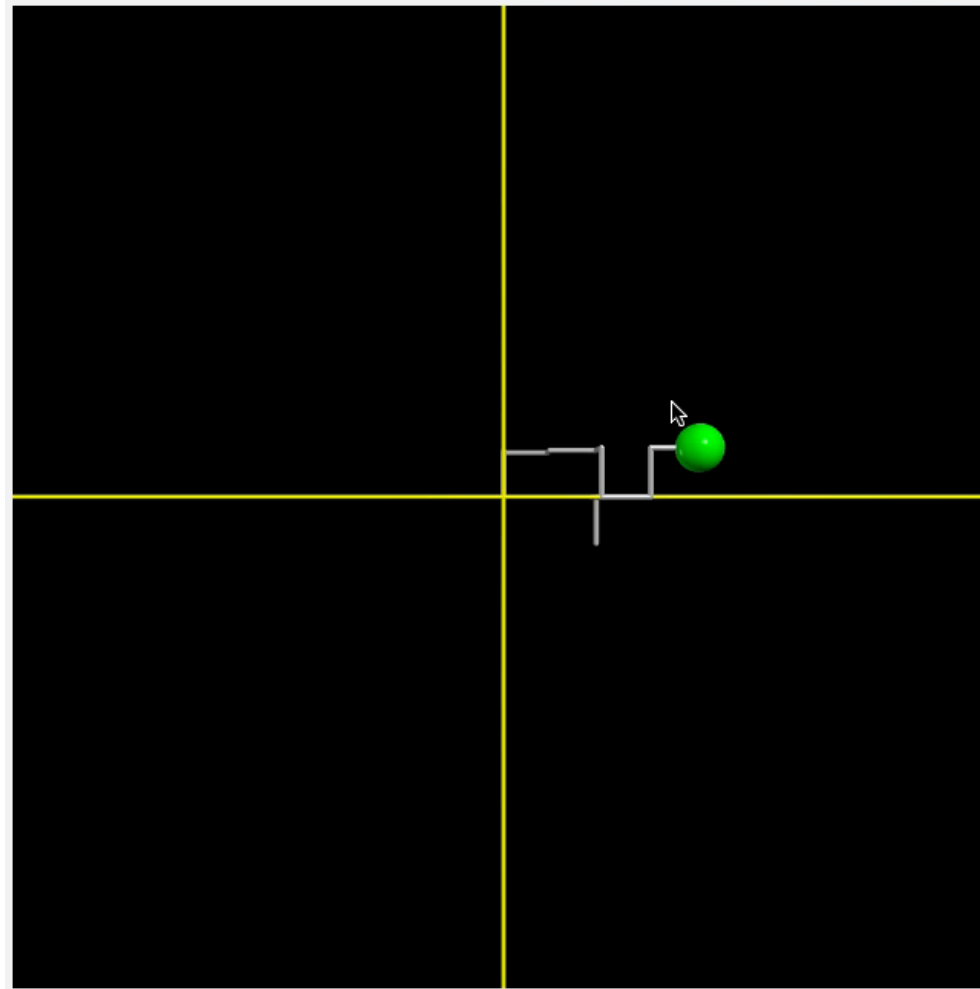
Camino aleatorio en 2 dimensiones



# Tema 7: Métodos estocásticos

## 7.2 Caminos aleatorios

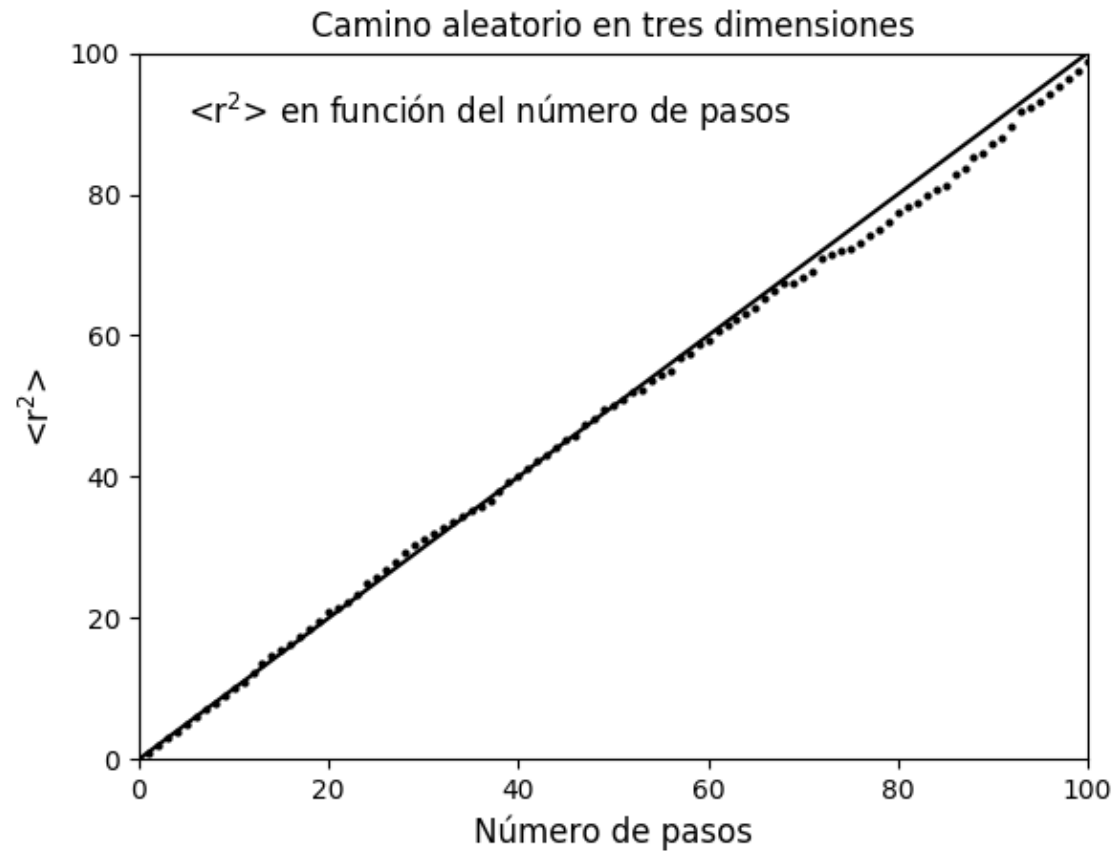
Camino aleatorio en 3 dimensiones



# Tema 7: Métodos estocásticos

## 7.2 Caminos aleatorios

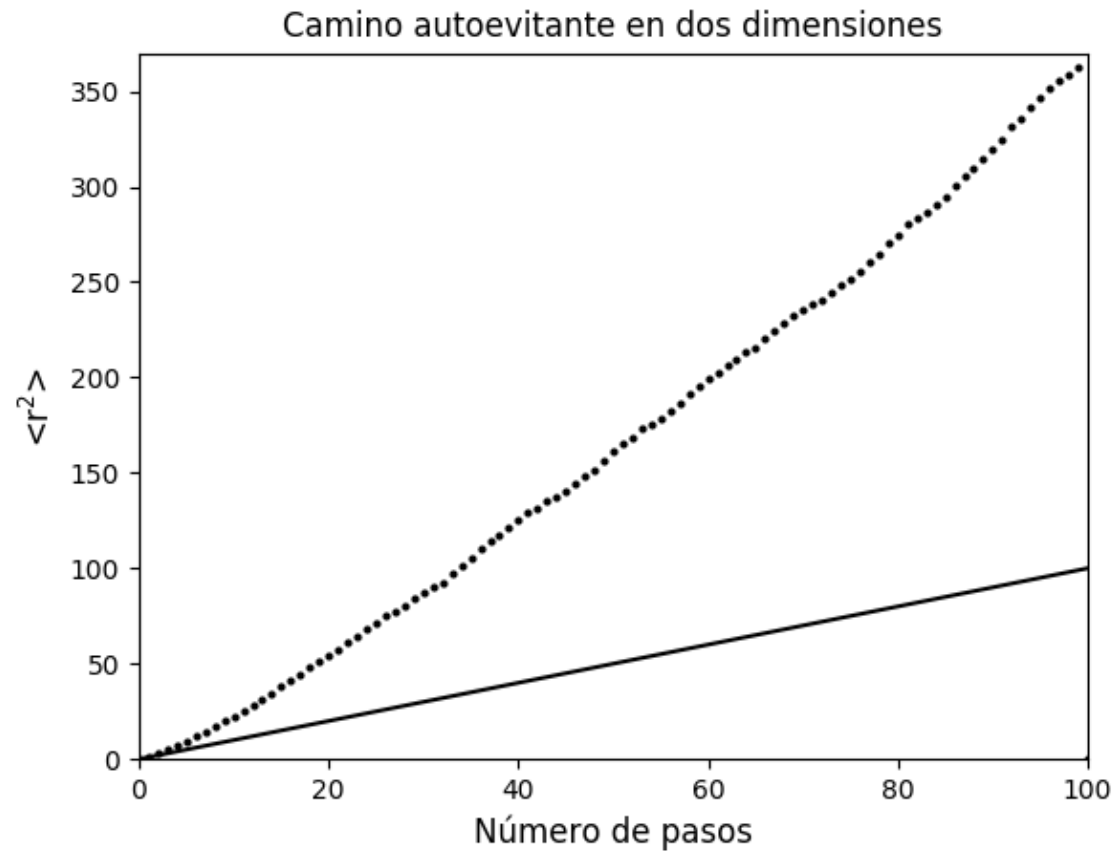
Camino aleatorio en 3 dimensiones



# Tema 7: Métodos estocásticos

## 7.2 Caminos aleatorios

Camino autoevitante en 2 dimensiones



# Tema 7: Métodos estocásticos

## 7.2 Caminos aleatorios

Camino aleatorio autoevitante en 2 dimensiones

