Problema 1: Generación de números pseudoaleatorios

El **método de congruencias lineales** es un algoritmo de generación de números aleatorios que funciona como una recurrencia a partir de un valor inicial x_0 denominado *semilla* y tres parámetros enteros a, c y M. Dichos números pseudoaleatorios están dados por

$$x_n = (ax_{n-1} + c) \bmod M, \qquad n = 1, 2, 3, \cdots$$

Como consecuencia de la operación modulo (residuo de la división entera), los números pseudoaleatorios generados están acotados entre 0 y $M - 1$. Una modificación de este algoritmo es que en la fórmula de recurrencia se utilice como semilla el *reloj* del procesador de la computadora y no el número pseudoaleatorio anterior x_{n-1}.

La implementación de este algoritmo es sencilla y se adjunta en el código `t4p1_pseudoaleatorios.py`. Algunos números generados con este método se muestran en la Figura 1.

Si se eligen los parámetros $a = 57$, $c = 1$, $M = 256$ y $x_0 = 10$ se genera una serie de números pseudoaleatorios que resulta ser **periódica**. Basta con revisar en que iteración del algoritmo se vuelve a generar el valor semilla x_0, es decir, determinar n tal que $x_n = x_0$; esto porque el funcionamiento de la recurrencia, cuando se mantienen los parámetros fijos, volvería a comenzar. Así, para los parámetros antes mencionados se determina que la secuencia generada tiene periodo 256, como se muestra en la Figura 1.

Existe un teorema llamado *teorema de Hull-Dobell* que establece las condiciones bajo las cuales este método de generación de números aleatorios muestra periodo M (como en el caso anterior). Cabe mencionar que la elección adecuada de los parámetros del método puede conducir a una buena generación de números aleatorios capaces de superar pruebas de aleatoriedad.

Una vez comprobada la existencia del periodo de la serie de números anteriores, se almacenaron los números generados anteriormente y se graficaron los puntos (x_{2i-1}, x_{2i}) con $i = 1, 2, \cdots, 128$. El resultado se muestra en la Figura 2. Adicionalmente se graficaron los puntos (i, x_i) con $i = 1, 2, \cdots, 256$, como se muestra en la Figura 3.

```
In [31]: runfile('/home/diego/Desktop/Fisica_Numerica/Tarea_4/
pseudoaleatorios.py', wdir='/home/diego/Desktop/
Fisica_Numerica/Tarea_4')
Algunos numeros pseudoaleatorios generados por el método de
congruencias lineales con parametros
a= 57 , c= 1 , M= 256 , seed= 12
[  1.  58. 235.  84. 181.  78.  95.  40. 233. 226.]


El periodo del método de congruencias lineales con parametros
a= 57 , c= 1 , M= 256 , seed= 10 es de:
 256
```

Figura 1. Generación de algunos números aleatorios por el método de congruencias lineales y determinación del periodo de la serie generada con los parámetros $(a, c, M, x_0) = (57, 1, 256, 10)$.

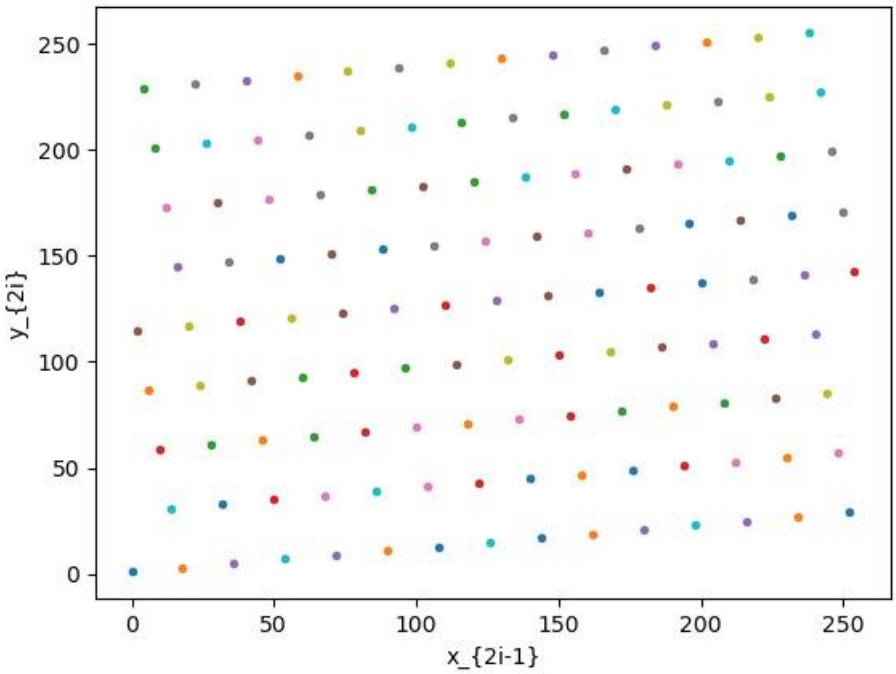

Figura 2. Gráfica de (x_{2i-1}, x_{2i}), $i = 1,2,\cdots,180$ utilizando la serie de números aleatorios generada por el método de congruencias lineales con $(a, c, M, x_0) = (57, 1, 256, 10)$.
Los pares de puntos muestran una distribución regular en el plano y se hace clara la acotación de los valores generados.

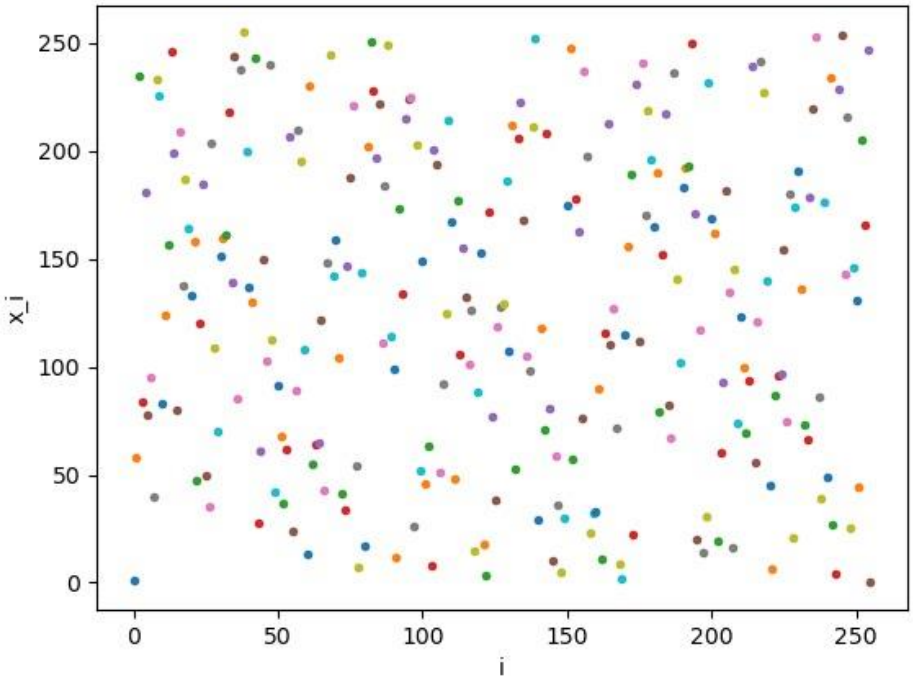

Figura 3. Gráfica de (i, x_i), $i = 1,2,\cdots,256$ utilizando la serie de números aleatorios generada por el método de congruencias lineales con $(a, c, M, x_0) = (57, 1, 256, 10)$.
La distribución espacial de los puntos, con énfasis en sus variaciones en el eje vertical, muestran la aleatoriedad con la que fueron generados.

Problema 2: Integración por el método de Montecarlo

Supóngase que se desea calcular la integral

$$\alpha = \int_0^1 g(x)\, dx$$

El valor esperado E de la función $g(x)$ con densidad de probabilidad $f(x)$ es

$$E\big(g(f)\big) = \int_{-\infty}^{\infty} g(x) f(x)\, dx$$

entonces la integral a calcular α se escribe como

$$\alpha = \int_0^1 g(x)\, dx = E\big(g(U)\big)$$

donde U es la **densidad de probabilidad uniforme** en el intervalo $[0,1]$.

Considérese ahora K variables aleatorias uniformes en $[0,1]$ independientes $U_1, U_2, \cdots, U_K$. Entonces las variables aleatorias en la función $g(x)$ son $g(U_1), g(U_2), \cdots, g(U_K)$ y también son independientes, tienen la misma distribución y la misma media α.

Utilizando la ley de los grandes números se tiene que

$$\lim_{K\to\infty} \frac{1}{K} \sum_{i=1}^{K} g(U_i) = E\big(g(U)\big) = \alpha$$

Esto proporciona un método para aproximar el valor de la integral $\int_0^1 g(x)\, dx$ por un método de Montecarlo utilizando números aleatorios uniformemente distribuidos en $[0,1]$.

$$\alpha = \int_0^1 g(x)\, dx = \lim_{K\to\infty} \frac{1}{K} \sum_{i=1}^{K} g(U_i) \approx \frac{1}{N} \sum_{i=1}^{N} g(U_i)\,, \qquad \text{con } N \text{ suficientemente grande}$$

Cabe mencionar que esta aproximación difiere del valor correcto de la integral por una cantidad ε que depende del N elegido.

Así, para estimar el valor de $\int_0^1 (1-x^2)^{\frac{3}{2}}\, dx$ se genera un número K suficientemente grande de números aleatorios U_i uniformemente distribuidos en $[0,1]$ y se realiza $\frac{1}{K}\sum_{i=1}^{K} g(U_i)$; esta última operación se realiza más convenientemente como $\sum_{i=1}^{N} \frac{g(U_i)}{N}$ en el algoritmo utilizado. Los resultados obtenidos se muestran en la Figura 4.

Para estimar el valor de $\int_{-2}^{2} \exp(x+x^2)\ dx$ se debe realizar un cambio de variable para transformar el dominio al intervalo $[0,1]$ y así utilizar el método desarrollado anteriormente. La elección de cambio adecuada es, definiendo $a = -2$ y $b = 2$:

$$y = \frac{x-a}{b-a} = \frac{x+2}{4} = \frac{1}{4}(x+2)$$

De donde

$$x = 4y - 2, \qquad dx = 4\,dy$$

haciendo que

$$\int_{-2}^{2} \exp(x + x^2)\; dx = \int_{0}^{1} 4\exp((4y-2) + (4y-2)^2)\;\; dy$$

Así, se implementa el mismo algoritmo que con la integral pasada y se estima el valor de la integral original. Los resultados obtenidos se muestran en la Figura 4.

Ambas implementaciones para aproximar las dos integrales se hacen en el programa `t4p2_integrales.py`.

```
In [25]: runfile('/home/diego/Desktop/Fisica_Numerica/Tarea_4/
integrales.py', wdir='/home/diego/Desktop/Fisica_Numerica/Tarea_4')
Se estima que el valor de la primera integral es:
 0.589066026282295
Se estima que el valor de la segunda integral es:
 93.6154467372938

In [26]: runfile('/home/diego/Desktop/Fisica_Numerica/Tarea_4/
integrales.py', wdir='/home/diego/Desktop/Fisica_Numerica/Tarea_4')
Se estima que el valor de la primera integral es:
 0.5883909663963596
Se estima que el valor de la segunda integral es:
 93.80047923301117

In [27]: runfile('/home/diego/Desktop/Fisica_Numerica/Tarea_4/
integrales.py', wdir='/home/diego/Desktop/Fisica_Numerica/Tarea_4')
Se estima que el valor de la primera integral es:
 0.590395555705641
Se estima que el valor de la segunda integral es:
 92.94065310086846
```

Figura 4. Estimaciones de integrales por el método de Montecarlo utilizando 100,000 números aleatorios uniformes en [0,1]. Las variaciones entre diferentes ejecuciones de los algoritmos se producen por la variación de los números aleatoriamente generados en cada caso, sin embargo, resultan útiles para acotar los valores numéricos de las integrales requisitadas.

Problema 3: Calculando a π

Para este problema, se utilizaron dos formas distintas de calcular el valor de π por métodos de Montecarlo:

- Por **razón de volúmenes**:
  Aquí se generan n puntos aleatorios $p_i = (x_i, y_i, z_i)$ con cada una de las coordenadas bajo una distribución uniforme en $[0,1]$, es decir, $0 \leq x_i, y_i, z_i \leq 1$.
  Luego, se realiza el reescalamiento de los n puntos que están en $[0,1]^3$ al cubo $[-1,1]^3$ utilizando un cambio de variable similar al del ejercicio anterior.
  Los volúmenes por considerar son el del cubo $[-1,1]^3$ (denominado V_c) y el de la esfera inscrita de radio 1 centrada en el origen, denominado V_e. Se sabe que $V_c = 8$ mientras que $V_e = \frac{4}{3}\pi$.
  La condición de los puntos p_i que están dentro de la esfera mencionada es que

$$x_i^2 + y_i^2 + z_i^2 \leq 1$$

  La razón de los puntos dentro de la esfera (digamos, un número n_e) entre los n puntos que están en el cubo resulta ser aproximadamente igual a la razón entre sus volúmenes respectivamente, es decir

$$\frac{n_e}{n} \approx \frac{V_e}{V_c} = \frac{\frac{4}{3}\pi}{8} = \frac{4\pi}{24} = \frac{\pi}{6}$$

  Por lo que, un valor aproximado de π, utilizando este método de Montecarlo (en el sentido de la generación de puntos aleatorios) es

$$\pi = 6\frac{n_e}{n}$$

  Este procedimiento se implementa en el código `t4p3_pi.py`. Los resultados obtenidos se muestran en la Figura 5.

- Por **estimación de integral**:
  En este procedimiento se implementa el procedimiento del ejercicio anterior para estimar el valor de una integral que tiene por resultado una expresión proporcional a π.
  Dada una circunferencia de radio 1 centrada en el origen del plano, se tiene la ecuación que la describe como

$$x^2 + y^2 = 1, \qquad -1 \leq x, y \leq 1$$

  de modo que la mitad superior de dicha circunferencia está descrita por la función

$$y(x) = \sqrt{1 - x^2}, \qquad -1 \leq x \leq 1$$

luego, el área bajo la curva representa el área de media circunferencia, la cual es $\frac{\pi}{2}$ en este caso. De modo que

$$\int_{-1}^{1} y(x)\,dx = \int_{-1}^{1} \sqrt{1-x^2}\,dx = \frac{\pi}{2}$$

La estimación del valor de la integral se puede realizar como el ejercicio anterior con el cambio de variable $u = \frac{1}{2}(x+1)$, de modo que

$$\frac{\pi}{2} = \int_{-1}^{1} \sqrt{1-x^2}\,dx = 2\int_{0}^{1} \sqrt{1-(2u-1)^2}\,du \approx \frac{2}{N}\sum_{i=1}^{N} g(U_i)$$

con $g(u) = \sqrt{1-(2u-1)^2}$ y U_i una variable aleatoria con distribución uniforme en [0,1], para un valor de N suficientemente grande, esto es

$$\pi \approx \frac{4}{N}\sum_{i=1}^{N} g(U_i)$$

Los resultados se muestran en la Figura 5 y también se implementan en el código `t4p3_pi.py`.

```
In [28]: runfile('/home/diego/Desktop/Fisica_Numerica/Tarea_4/
untitled4.py', wdir='/home/diego/Desktop/Fisica_Numerica/Tarea_4')
El calculo de pi por razon de volumenes por Montecarlo con  1000000
puntos es de:
3.13695
El calculo de pi por aproximacion de integral por Montercalo con
1000000  numeros aleatorios es de:
3.1405977996714394

In [29]: runfile('/home/diego/Desktop/Fisica_Numerica/Tarea_4/
untitled4.py', wdir='/home/diego/Desktop/Fisica_Numerica/Tarea_4')
El calculo de pi por razon de volumenes por Montecarlo con  1000000
puntos es de:
3.137442
El calculo de pi por aproximacion de integral por Montercalo con
1000000  numeros aleatorios es de:
3.1391658164714 99

In [30]: runfile('/home/diego/Desktop/Fisica_Numerica/Tarea_4/
untitled4.py', wdir='/home/diego/Desktop/Fisica_Numerica/Tarea_4')
El calculo de pi por razon de volumenes por Montecarlo con  1000000
puntos es de:
3.144234
El calculo de pi por aproximacion de integral por Montercalo con
1000000  numeros aleatorios es de:
3.1426320487609862
```

Figura 5. Estimaciones del valor de π a través de dos métodos de Montecarlo distintos.
Se puede observar que, a pesar de la gran cantidad de números aleatorios generados para el cálculo en cada método, no se obtiene un valor de π correcto en más allá que las primeras dos cifras decimales, lo que evidencia la desventaja del uso de este tipo de métodos en contraste con su facilidad de implementación.
Cabe mencionar que, en cada ejecución del programa, los resultados varían como consecuencia de la variación de los números utilizados en cada caso.