Informe 05 - Resolución numérica de la ecuación de Poisson 2D

Profesor: Valentino González
Auxiliar: Felipe Pesce
Integrantes: Nicolás Troncoso[1]
[1] *Universidad de Chile, Facultad de Ciencias Físicas y Matemáticas, Departamento de Física.*

I. INTRODUCCIÓN

En el siguiente informe se presenta implementación y resultados de dos algoritmos que basan su funcionamiento en generación de datos de manera aleatoria: Metropolis y Montecarlo.

La motivación principal de este trabajo es aprender utilizar ambos algoritmos para aplicarlos a situaciones sencillas y estudiar su funcionamiento

En la sección **II** se presentan los algoritmo a utilizar y su implementación. En la sección **III** se presentan dos casos específicos donde serán utilizados. En la sección **IV** se muestran los resultados obtenidos. En la sección **V** se discutirán los resultados y se concluirá sobre ellos.

II. ALGORITMO

A. Montecarlo

El algoritmo Montecarlo Simple consiste en aproximar la integral de $f(\vec{x})$ función escalar mediante la evaluación de la función en un número elevado de puntos $\vec{x}_0$ distribuidos de manera uniforme en el dominio y generados de manera aleatoria. El dominio sobre el cual se generan los puntos corresponde al mínimo conjunto posible que contenga la integral a calcular, en el caso de 3D, corresponde a un cubo que tenga por subconjunto el cuerpo a integrar. La aproximación consiste en:

$$I = \frac{V}{N}\sum_{i=1}^{N} f(\vec{x}_i) \tag{II.1}$$

B. Metropolis

El algoritmo de Metropolis se utiliza para generar conjuntos de puntos distribuidos según una función $W(x)$. El algoritmo utiliza un punto inicial como semilla x_n y en base a una distancia de paso d y al criterio de Metropolis decide si un punto $x_p \in [x_n - d, x_n + d]$ generado de manera aleatoria se agrega al conjunto o si en cambio se vuelve a agregar el punto x_n.

El criterio de Metropolis consiste en evaluar en la función $W(x)$ los puntos x_n y x_p y verificar si su cociente es mayor que un número r generado de manera aleatoria según una distribución uniforme $U(0,1)$. Luego, si es que se tiene:

$$\frac{W(x_p)}{W(x_n)} > r \tag{II.2}$$

Entonces x_p se agrega al conjunto generado, en caso contrario se agrega x_n. Este paso debe repetirse N veces utilizando como semilla el punto aceptado en el paso anterior. El problema de Metropolis es que debe estudiarse la forma de la función para poder encontrar x_n inicial y d óptimos tal que la proporción entre puntos rechazados por el criterio de Metropolis y los puntos aceptados sea aproximadamente $1:1$.

III. CASOS A ESTUDIAR

A. Sólidos Intersectados

Tenemos la intersección de dos sólidos, un Toroide:

$$z^2 + \left(\sqrt{x^2+y^2} - 3\right) \leq 1, \tag{III.1}$$

y un cilindro:

$$(x-2)^2 + z^2 \leq 1. \tag{III.2}$$

Además, la densidad de este sólido varía según:

$$\rho(x,y,z) = \frac{1}{2}\cdot(x^2+y^2+z^2). \tag{III.3}$$

El objetivo de este problema es encontrar la masa del sólido resultante de la intersección y la posición del centro de masa. Para ello se utilizará Montecarlo Simple y se calculará $\vec{R}_{CM}$ mediante:

$$R_{CM} = \frac{\sum \vec{r}_i \rho(\vec{r}_i)}{\sum \rho(\vec{r}_i)} \tag{III.4}$$

Debido a que se considera un $dm = \rho dV$ para cada punto, los dV son iguales y lo podemos quitar de la ecuación del centro de masa. Los $\vec{r}_i$ utilizados para calcular la masa y el centro de masa son generados de manera aleatoria siguiendo una distribución uniforme para cada coordenada (x,y,z).

Un análisis simple de la forma de la figura resultante de la intersección permite determinar que el cubo minimal que contiene a la figura viene dado por:

$$\begin{aligned} x &\in [1,3] \\ y &\in [-4,4] \\ z &\in [-1,1] \end{aligned} \tag{III.5}$$

B. Muestra Aleatoria

Se desea obtener una muestra aleatoria de números que sigan la función distribución $W(x)$:

$$W(x) = \frac{7}{2} e^{\frac{-(x-3)^2}{3}} + 2e^{-2(x+1{,}5)^2}. \qquad \text{(III.6)}$$

Para ello se utilizó el algoritmo de Metropolis para generar distintos conjuntos de datos.

IV. RESULTADOS

A. Montecarlo Simple

Para la integración por Montecarlo Simple de la intersección del toroide con el cilindro se obtuvieron los resultados mostrados en el cuadro I

Iteración	1	2	3	4
n	10^2	10^3	10^2	1
N	10^5	10^5	10^6	10^8
Masa	70.9490	70.9711	70.9711	70.9711
σ_m	0.2665	0.2572	0.0871	-
CM_x	2.0803	2.0802	2.0802	2.0802
σ_x	0.0023	0.0024	0.0007	-
CM_y	-0.0001	0.0007	0.0007	0.0007
σ_y	0.0120	0.0117	0.0033	-
CM_z	-0.0003	0.0000	0.0000	0.0000
σ_z	0.0021	0.0022	0.0007	-
Tiempo	$\approx$ 3 min	$\approx$ 30 min	$\approx$ 25 min	$\approx$ 90 min

Cuadro I: En el cuadro, n representa el número de veces que se integro mediante Montecarlo simple utilizando N puntos. Tanto la masa como las posiciones del centro de masa se muestran con su error estándar σ. Para la iteración 4 no hay errores medidos pues sólo se iteró una vez.

B. Metropolis

En primer lugar, se busco el paso d tal que los puntos aceptados y rechazados fuesen aproximadamente iguales, es decir, se obtuviese un $\approx 50\,\%$ de puntos aceptados en total. Se midió el porcentaje de puntos aceptados para distintos d y se obtuvo el gráfico de la Figura 1, donde cada iteración fue calculada utilizando $N = 10^5$ puntos.

Este ejercicio permitió determinar que un d óptimo era $d = 3{,}96$, que se utilizó en adelante para los cálculos. La Figura 2 muestra el histograma obtenido para el conjunto de datos generados utilizando $N = 10^8$. Para normalizar se utilizó la integral numérica de $W(x)$ entre -100 y 100. Un cálculo más preciso hubiese utilizado la integral analítica de $x \in [-\infty, \infty]$, sin embargo debido al rápido decaimiento que la exponencial presenta, no se considero necesario un cálculo más elaborado.

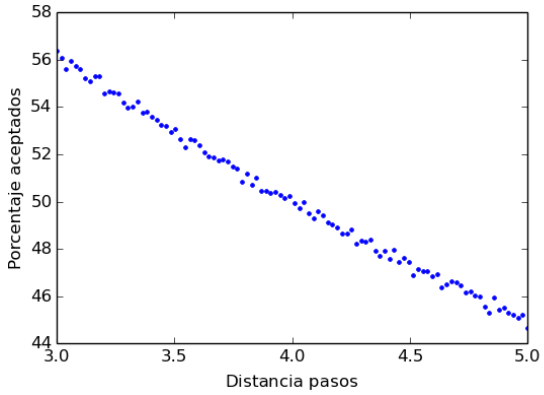

Figura 1: Porcentaje de valores aceptados en función del valor de la distancia del paso d.

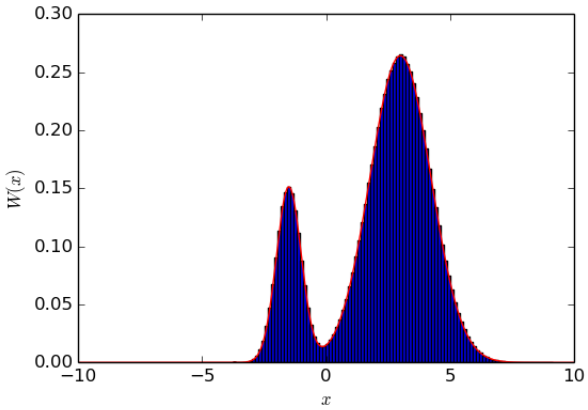

Figura 2: Las barras en azul representan el histograma obtenido para $N = 10^8$ datos generados con el algoritmo de Metropolis utilizando la versión normalizada de $W(x)$. El histograma se genero con 100 bins entre -10 y 10. En rojo la versión normalizada de $W(x)$.

Finalmente se generaron 100 conjuntos diferentes de $N = 10^6$ elementos cada uno, distribuidos según $W(x)$ utilizando el algoritmo de Metropolis. Se calculo el histograma normalizado de cada uno de estos conjuntos utilizado 100 bins del mismo ancho entre -10 y 10, para luego calcular el error estándar σ de cada bin utilizando la función `numpy.std()`. Los resultados obtenidos se muestran en la Figura 3.

V. DISCUSIÓN Y CONCLUSIÓN

Para el algoritmo de Montecarlo simple podemos observar que la combinación de número de iteraciones y número de puntos utilizados permite encontrar soluciones más óptimas que simplemente aumentar el número de puntos utilizados en el sentido de correr programas en menor cantidad de tiempo y con mayor precisión.

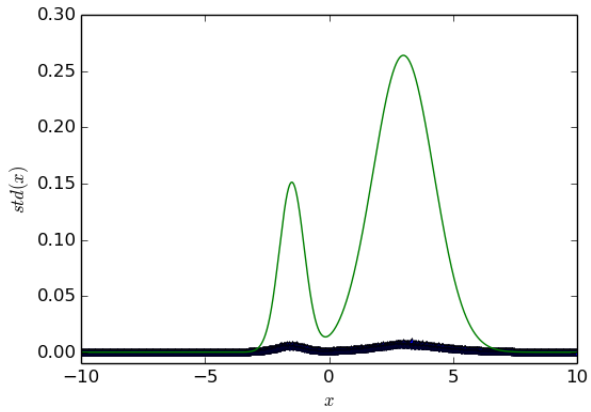

Figura 3: En verde la función $W(x)$. Los puntos sólidos en negro indican el error std obtenido para cada bin.

En Metropolis, encontramos cierta dispersión en el gráfico de la Figura 1, lo cual se atribuye a la propia naturaleza azarosa del método, sin embargo es clara la tendencia de disminuir el porcentaje de puntos aceptados a medida que se aumenta d. Para un d óptimo tal que los puntos aceptados son aproximadamente los mismos que los rechazados por el criterio de Metropolis, encontramos que el conjunto generado por Metropolis sigue el mismo comportamiento que la función de distribución utilizada, tal y como se puede ver en la Figura 2.

Un análisis de el error estándar asociado a los bins de cada conjunto permite observar que mientras mayor sea el valor de $W(x)$, mayor será también el error asociado al bin, tal y como se muestra en la Figura 3.

Como conclusión general, se resalta la capacidad de ambos algoritmos de resolver problemas mediante métodos de generación azarosa de puntos de manera mucho más simple que haberlo resuelto de maneras tradicionales, tal como generar un renderizado del cubo minimal en el caso de la intersección de los sólidos y evaluar punto a punto.