Ayudantía IV: Preparando la PEP I, Métodos Numéricos y Errores Computacionales

Física Computacional III — Licenciatura en Astrofísica con mención en Ciencia de Datos

Profesor: Omar Fernández
Ayudante: Nicolás Campos

22 de abril de 2025

Contexto

Sea la función $f(x) = e^{-x^2}$, una forma simple de la función gaussiana que se presenta en múltiples contextos físicos. Estudiaremos su comportamiento en un entorno computacional, considerando representación numérica, derivación, y resolución de ecuaciones no lineales.

Parte A: Representación numérica y errores

1. Calcula en Python el valor de $f(0{,}1)$ usando `float32` y `float64`. ¿Qué observas?
2. (Calcula el **error relativo** entre ambas representaciones. ¿Qué implica esto sobre la precisión numérica en Python?
3. ¿Por qué es importante tener conciencia sobre el tipo de precisión cuando se modelan sistemas físicos o se acumulan operaciones sucesivas?

Parte B: Derivación numérica

1. Usando diferencias hacia adelante y centradas, estima $f'(1)$ con un paso $h = 0{,}1$.
2. Compara tus resultados con la derivada analítica: $f'(x) = -2xe^{-x^2}$. Calcula errores absolutos y relativos.
3. Evalúa $f'(1)$ con diferencias centradas para $h = 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}$. Grafica el error relativo en función de h. ¿Qué observas?
4. ¿Por qué no conviene elegir un h extremadamente pequeño? ¿Qué efectos computacionales pueden surgir?

Parte C: Solución de ecuaciones no lineales

1. Resuelve la ecuación $e^{-x^2} = 0{,}5$ usando el **método de bisección** en $x \in [0, 1]$ con tolerancia $\epsilon = 10^{-5}$.

2. Resuelve el mismo problema con el **método de Newton-Raphson** partiendo desde $x_0 = 0{,}5$. Usa la derivada exacta.

3. ¿Por qué es **crucial definir un criterio de tolerancia** ϵ en métodos iterativos? ¿Qué pasa si es muy grande o muy pequeña?

Parte D: Reflexión computacional y buenas prácticas

1. ¿Qué otras **consideraciones computacionales** debemos tener al aplicar métodos numéricos en física? Menciona al menos dos.

2. ¿Por qué no basta con obtener un resultado que "parece correcto"? ¿Cómo se puede **validar** un cálculo numérico?