Taller de Física Computacional 2020

Primer Parcial

29 de octubre de 2020

IMPORTANTE:

- Resuelva los ejercicios en un notebook nuevo que deberá subir como respuesta a esta tarea una vez resuelto.
- Comente el código de manera que sea legible por otra persona.
- Todas las celdas deben ejecutar sin error. Se recomienda hacer un reinicio del núcleo y evaluar todas las celdas en orden para asegurarse que ejecuta como corresponde antes de entregar.
- Lea **atentamente** las consignas.

Ejercicio 1 (*20 puntos*) Graficar la función

$$f(x) = e^{-x} \sin 10x$$

en el intervalo $[0, 10]$.

Ejercicio 2 (*20 puntos*) Implementar una función que clasifique un triángulo como rectángulo, acutángulo u obtusángulo dadas las longitudes de sus lados a, b y c (números reales) en base a las condiciones enumeradas en la siguiente tabla:

Condición	Tipo
si $a^2 + b^2 = c^2$	el triángulo es **rectáncgulo**
si $a^2 + b^2 > c^2$	el triángulo es **acutángulo**
si $a^2 + b^2 < c^2$	el triángulo es **obtusángulo**

Ejercicio 3 (*20 puntos*) Implementar una función que determine las componentes del vector **C**, donde $\mathbf{C} = \mathbf{A} \times \mathbf{B}$ es el producto vectorial entre los vectores **A** y **B**. Nota: usar `numpy.cross` o similar es **trampa**.

Ejercicio 4 (*20 puntos*) La ecuación Pitagórica

$$i^2 + j^2 = k^2$$

tiene un número infinito de soluciones enteras **positivas** llamadas "triplas Pitagóricas".

a (*10 puntos*) Diseñar e implementar una función que calcule (imprima) todas las triplas Pitagóricas para un dado m tal que $i \leq m$, $j \leq m$ y $k \leq m$. No importa si el código devuelve algunas triplas repetidas.

b (*10 puntos*) Modificar la función para "probar" que el último teorema de Fermat, que asegura que no existe ninguna solución de enteros positivos a

$$i^n + j^n = k^n,$$

es válido hasta un cierto entero positivo m.

c (**opcional**) Modificar la función del ítem (a) para que devuelva una secuencia de tuplas que contenga las triplas únicas.

Ejercicio 5 (*20 puntos*) Una serie de Fourier tiene la forma

$$s(x) = \frac{a_0}{2} + \sum_{i=1}^{N} \left(a_n \cos(nx) + b_n \sin(nx)\right)$$

y puede aproximar (con valores adecuados de $\{a_n\}$ y $\{b_n\}$) cualquier función periódica de período 2π.

a) (*8 puntos*) Implementar una función que acepte como parámetros dos listas conteniendo los conjuntos de coeficientes $\{a_n\}$ y $\{b_n\}$ y evalúe la correspondiente serie de Fourier para un argumento $x \in \mathbb{R}$.

b) (*8 puntos*) Utilizar la función anterior para implementar una nueva función que evalúe la serie de Fourier de la función *serrucho* para la cual

$$a_n = 0 \; \forall \; n$$

y

$$b_n = \frac{2(-1)^{n+1}}{n\pi} \; \forall \; n \geq 1$$

c) (*4 puntos*) Comparar gráficamente la aproximación de órden 10 para la función serrucho. Utilizar la siguiente definición para la función serrucho:

```
def f(x):
    return 2*((x+1/2) - math.floor(x+1/2)) - 1
```

Nota: En esta definición $f(x)$ es periódica con período 1.