Cristóbal Espinoza Latorre
202273565-1

Pregunta 1

a) Lógica: Determinamos el valor de c

Tengamos el valor de c_i como $\frac{x + x_0}{2}$

Tengamos los primeros K términos de la expansión como $T_K(x)$

$\hookrightarrow f(x) = T_K(x) + \frac{f^{(K+1)}(c)}{(K+1)!}(x - x_0)^{K+1}$

Despejemos $f^{(K+1)}(c) = \underbrace{\frac{(f(x) - T_K(x))(K+1)!}{(x - x_0)^{K+1}}}_{\phi(x) \text{ por simplicidad}}$

Entonces podemos tener

$$f^{(K+1)}(c) = \phi(x) \text{, y al restarlos: } G(c) = f^{(K+1)}(c) - \phi(x) = 0$$

Y con eso iteramos en newton.

Algoritmo: 1. Primero obtenemos el valor $f(x)$.

2. Luego $T_K(x)$, es decir, las derivadas, potencias y factoriales en una sumatoria de 0 hasta K.
3. Calcular $\phi(x)$.
4. Definir $G(c) := f^{(K+1)}(c) - \phi(c) = 0$
5. Aplicar newton y obtener la aproximación de c.

b)
```
def find_c(x, x0, k):
    dx = x-x0

    # Paso 1
    fx = K_derivatives_of_f(x,0)

    # Paso 2
    Vector_K = np.arange(K+1)
    derivadas = K_derivatives_of_f(x0, vector_k)
    Powers = np.power(vector_k)
    factoriales = my_factorial(vector_k)
    Tk = np.dot(derivadas/factoriales, Powers)
```

```
Phi = (my_factorial(K+1) * (fx - Tk)/np.power(dx, K+1)

#Paso 4
C_i = (X0 + X)/2
def G(c):
    return K_derivative_of_f(C, K+1) - Phi
def Gp(c):
    return K_derivative_of_f(C, K+2)

C = Newton1D(G, Gp, C_i, M=1)
return C.
```

Pregunta 2.

a) El producto punto da 2^{-1499}, pero como estamos en double precision, el resultado es $dot = 0$

n_x es obtenido con my_norm2:

$$max\_abs\_value = 2^{-809}$$

$$X\_tilde = \frac{[2^{-809}, 2^{-809}]}{2^{-800}} = [1, 1]$$

$$X\_tilde\_norm = \sqrt{1^2 + 1^2} = \sqrt{2}$$

finalmente norm_2 es: $X\_tilde\_norm * max\_abs\_value = 2^{-800}\sqrt{2}$

$$n\_X = 2^{-800}\sqrt{2}$$

Luego $dot / n\_x = 0$ ya que $dot = 0$

Si seguimos, vemos claramente que, como $out = 0$, el resultado final es 0.

b)

a-b) Para prevenir que $dot = 0$, normalizamos a X y a Y antes de evaluar el producto punto haciendo uso de

$$\|X\| = |\alpha| \left\| \frac{X}{\alpha} \right\|$$ con α el valor máximo de X

entonces $\dfrac{\langle X, Y\rangle}{\alpha\|\frac{X}{\alpha}\|_2\ \beta\|\frac{Y}{\beta}\|_2}$ "Usamos" esos α y β y:

$$\frac{\langle \frac{X}{\alpha}, \frac{Y}{\beta}\rangle}{\|\frac{X}{\alpha}\|_2\ \|\frac{Y}{\beta}\|_2} \Rightarrow \frac{\langle X_n, Y_n\rangle}{\|X_n\|_2\ \|Y_n\|}$$

Pasos:

1. Escalar a X y a Y $\Rightarrow$ $X_n = \frac{X}{\alpha}$, $Y_n = \frac{Y}{\beta}$
2. Calcular el producto punto de $\langle X_n, Y_n\rangle$
3. Calcular la norma$_2$ de X_n y de Y_n
4. Hacer la división.

b.b) Evaluando: $X_n = [1, 1]$, $Y_n = [1, 1]$

$$\frac{\langle X_n, Y_n\rangle}{\|X_n\|_2\ \|Y_n\|_2} = \frac{2}{\sqrt{2}\ \sqrt{2}} = \frac{2}{2} = 1 \ :D$$

c)
```
def cosine_simircrity_v2(X,Y):
    X_max = np.max(np.abs(X))
    X_n = X/X_max
    Y_max = np.max(np.abs(Y))
    Y_n = Y/Y_max

    dot = np.dot(X_n, Y_n)

    n_x = norma_2_direct(X_n)
    out = dot/n_x

    n_Y = norma_2_direct(Y_n)

    CS = out/n_Y
    return CS
```