1.1 Utilice las fórmulas de diferencias finitas de dos puntos adelantadas y atrasadas en los puntos indiciados (y utilizando solo los puntos indicados) en las siguientes funciones. Compare con los valores exactos.

a) $f(x) = 2\cos(2x) - x$, con $x_0 = -0.3$, $x_1 = -0.2$, y $x_2 = -0.1$.

b) $f(x) = x^2 \ln x + 1$, con $x_0 = 1.0$, $x_1 = 1.2$, y $x_2 = 1.4$.

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Como solo tenemos $f(x_n)$: $x_n = \{x_0, x_1, x_2\}$

Tomamos como h a la distancia entre x_n y x_{n+1}

$\Rightarrow h = x_{n+1} - x_n$

Planteamos ambos casos

lo mismo aplica a

$h = x_n - x_{n-1}$

<u>Adelantada:</u>

$$f'(x) = \frac{f(\overbrace{x_{i+1}}^{x_i + h}) - f(x_i)}{\underbrace{x_{i+1} - x_i}_{h}}$$

<u>Atrasada:</u>

$$f'(x) = \frac{f(x_i) - f(\overbrace{x_{i-1}}^{x_i - h})}{\underbrace{x_i - x_{i-1}}_{h}}$$

```python
def fA(x):
    return 2*np.cos(2*x) - x

def fB(x):
    return x**2 * np.log(x) + 1

def derivadaAprox(f, xn0, xn1):
    return (f(xn1) - f(xn0)) / (xn1 - xn0)

fs = [fA, fB]
x0s = [[-0.3, -0.2, -0.1], [1.0, 1.2 , 1.4]]

for i in range(2):
    f = fs[i]
    x0 = x0s[i]
    h = x0[i] - x0[i+1]

    for j in range(len(x0) - 1):
        print(f"Derivada adelantada en x = {x0[j]}: {derivadaAprox(f, x0[j], x0[j + 1])}")

    for j in range(1, len(x0)):
        print(f"Derivada atrasada en x = {x0[j]}: {derivadaAprox(f, x0[j - 1], x0[j])}")
    print()
```

```
Derivada adelantada en x = -0.3: 0.914507581864137
Derivada adelantada en x = -0.2: 0.18011167676712958
Derivada atrasada en x = -0.2: 0.914507581864137
Derivada atrasada en x = -0.1: 0.18011167676712958

Derivada adelantada en x = 1.0: 1.312715208916473
Derivada adelantada en x = 1.2: 1.9847127099714128
Derivada atrasada en x = 1.2: 1.312715208916473
Derivada atrasada en x = 1.4: 1.9847127099714128
```

1.2 Utilice las fórmulas de diferencias finitas de tres puntos más conveniente en los puntos indiciados (y utilizando solo los puntos indicados) en las siguientes funciones. Compare con los valores exactos.

a) $f(x) = e^{2x} - \cos(2x)$, con $x_0 = -0.3$, $x_1 = -0.2$, $x_2 = -0.1$, y $x_3 = 0$.

b) $f(x) = x \sin x + x^2 \cos x$, con $x_0 = 1.1$, $x_1 = 1.2$, $x_2 = 1.3$, y $x_3 = 1.4$.

Con extremo:

con $c \in (x_0, x_2)$

$$f'(x_0) = \frac{1}{2h}\left(-3f(x_0) + 4f(\underbrace{x_1}_{x_0+h}) - f(\underbrace{x_2}_{x_0+2h})\right) + \frac{h^2}{3}f'''(c)$$

Con punto medio:

con $c \in (x_0, x_2)$

$$f'(x_0) = \frac{1}{2h}\left(f(\underbrace{x_1}_{x_0+h}) - f(x_0 - h)\right) - \frac{h^2}{6}f'''(c)$$

```python
def fA(x):
    return np.exp(2 ** x) - np.cos(2 * x)

def fB(x):
    return x * np.sin(x) + x**2 * np.cos(x)

def extremo(f, x0, h):
    return (1/(2 * h)) * (-3 * f(x0) + 4 * f(x0 + h) - f(x0 + 2 * h))

def puntoMedio(f, x0, h):
    return (1/(2 * h)) * (f(x0 + h) - f(x0 - h))

fs = [fA, fB]
x0s = [[-0.3, -0.2, -0.1, 0], [1.1, 1.2, 1.3, 1.4]]

for i in range(2):
    f = fs[i]
    x0 = x0s[i]
    h = x0[i] - x0[i+1]

    print(f"Derivada en extremo en x = {x0[0]}: {extremo(f, x0[0], h)}")

    for j in range(1, len(x0) - 1):
        print(f"Derivada en punto medio en x = {x0[j]}: {puntoMedio(f, x0[j], h)}")
    print()
```

```
Derivada en extremo en x = -0.3: 0.11378565322984092
Derivada en punto medio en x = -0.2: 0.6725007801355666
Derivada en punto medio en x = -0.1: 1.2555873080202098

Derivada en extremo en x = 1.1: 1.3447166961870594
Derivada en punto medio en x = 1.2: 0.8775962921268802
Derivada en punto medio en x = 1.3: 0.3626157623061751
```

→ Primero uso extremo porq no tenemos x_{-1}

↳ Usamos PM porq es mejor

Para x_3 no hacemos nada porque no hay x_4

1.3 Dado un circuito con voltaje $V(t)$, inductancia L, resistencia R y corriente $I(t)$, la primera ley de Kirchhoff nos dice que

$$V(t) = L\frac{dI}{dt} + RI,$$

Suponiendo que se mide $I(1.0) = 3.10$, $I(1.01) = 3.12$, $I(1.02) = 3.14$, $I(1.03) = 3.18$ e $I(1.04) = 3.24$, y que $L = 0.98$ y $R = 0.142$. Calcule $V(t)$ en los tiempos dados.

Aproximamos $\frac{\partial I}{\partial t}(t)$ con los metodos:

Para $t = 1 \rightarrow$ 3 puntos extremo

Para $t = \{1,01, 1,02, 1,03\} \rightarrow$ 3 puntos medio

Para $t = 1,04 \rightarrow$ dos puntos atrasado

```python
t = [1.0, 1.01, 1.02, 1.03, 1.04]
L = 0.98
R = 0.142
h=0.01

def I(t0):
    return [3.1, 3.12, 3.14, 3.18, 2.24][t.index(t0)]

def extremo(f, x0, h):
    return (1/(2 * h)) * (-3 * f(x0) + 4 * f(x0 + h) - f(x0 + 2 * h))

def puntoMedio(f, x0, h):
    return (1/(2 * h)) * (f(x0 + h) - f(x0 - h))

def derivadaAprox(f, xn0, xn1):
    return (f(xn1) - f(xn0)) / (xn1 - xn0)

def V(I, dI, L, R):
    return L * dI + R * I

dI = []
dI.append(extremo(I, t[0], h))
for i in range(1, 4):
    dI.append(puntoMedio(I, t[i], h))
dI.append(derivadaAprox(I, t[4], t[3]))

for i in range(5):
    print(f"V({t[i]}) = {V(I(t[i]), dI[i], L, R)}")
```

```
V(1.0) = 2.40019999999998
V(1.01) = 2.4030400000000016
V(1.02) = 3.3858800000000024
V(1.03) = -43.648439999999994
V(1.04) = -91.80191999999992
```

1.4 Calcule $f'(x)$ para $f(x) = -0.4\tanh(50x) + 0.6$ con $x \in [-1, 1]$. Pruebe numéricamente en *python*con distintos esquemas y pasos, comparando con la derivada exacta.

Para $x \in [-1, 1]$ probemos $h = \{0{,}5,\ 0{,}25,\ 0{,}1\}$

Probamos cada metodo en su intervalo posible

```python
A, B = (-1, 1)
h = [0.5]
x0 = A

def f(x):
    return -0.4 * np.tanh(50 * x) + 0.6

# La de dos puntos
def derivadaAprox(f, xn0, xn1):
    return (f(xn1) - f(xn0)) / (xn1 - xn0)

# Las de 3 puntos
def extremo(f, x0, h):
    return (1/(2 * h)) * (-3 * f(x0) + 4 * f(x0 + h) - f(x0 + 2 * h))

def puntoMedio(f, x0, h):
    return (1/(2 * h)) * (f(x0 + h) - f(x0 - h))

for i in range(len(h)):
    print(f"Probamos con h = {h[i]}")

    # los whiles estan hechos muy rusticos pero es para que salgan mejor los prints
    print("Metodo de dos puntos adelantada en:")
    while True:
        # Dos puntos adelantada
        if x0 - h[i] >= A:
            print(f"x = {x0}: {derivadaAprox(f, x0, x0 - h[i])}")

        if x0 + h[i] > B:
            break
        x0 += h[i]

    x0 = A

    print("Metodo de dos puntos atrasada en:")
    while True:
        # Dos puntos atrasada
        if x0 + h[i] <= B:
            print(f"x = {x0}: {derivadaAprox(f, x0 + h[i], x0)}")

        if x0 + h[i] > B:
            break
        x0 += h[i]

    x0 = A

    print("Metodo de tres puntos en extremo en:")
    while True:
        # Tres puntos en extremo
        if x0 + 2 * h[i] <= B:
            print(f"x = {x0}: {extremo(f, x0, h[i])}")

        if x0 + h[i] > B:
            break
        x0 += h[i]

    x0 = A

    print("Metodo de tres puntos en punto medio en:")
    while True:
        # Tres puntos en punto medio
        if x0 - h[i] >= A and x0 + h[i] <= B:
            print(f"x = {x0}: {puntoMedio(f, x0, h[i])}")

        if x0 + h[i] > B:
            break
        x0 += h[i]

    x0 = A
    if i < 2:
        print("=====================================================")
```

```
Probamos con h = 0.5
Metodo de dos puntos adelantada en:
x = -0.5: -0.0
x = 0.0: -0.8
x = 0.5: -0.8
x = 1.0: -0.0
Metodo de dos puntos atrasada en:
x = -1: -0.0
x = -0.5: -0.8
x = 0.0: -0.8
x = 0.5: -0.0
Metodo de tres puntos en extremo en:
x = -1: 0.4
x = -0.5: -0.8
x = 0.0: -1.2
Metodo de tres puntos en punto medio en:
x = -0.5: -0.4
x = 0.0: -0.8
x = 0.5: -0.4
```

Se puede hacer con h mas chicas pero la terminal se vuelve muy larga

1.5 Dado $h > 0$, considere las siguientes fórmulas de diferencias finitas para aproximar la derivada de una función $u(x)$:

$$D_+u(x) = \frac{u(x+h) - u(x)}{h}, \quad D_-u(x) = \frac{u(x) - u(x-h)}{h}, \quad D_0u(x) = \frac{u(x+h) - u(x-h)}{2h},$$

$$D_3u(x) = \frac{1}{6h}\left[2u(x+h) + 3u(x) - 6u(x-h) + u(x-2h)\right].$$

Utilizando desarrollos de Taylor, justifique que estas expresiones aproximan la derivada $u'(x)$ con errores de orden $\mathcal{O}(h^p)$, para $p = 1, 1, 2, 3$ respectivamente.

Para la función $u(x) = \sin(x)$, grafique en escala log-log el error de cada método al evaluar en $x = 1$, para $h = 10^{-1}, 5 \cdot 10^{-2}, 10^{-2}, \ldots, 10^{-6}$.

a)

Definimos el error como

$$E = \left| u'(x) - D_+u(x) \right|$$

Taylor de $u(x+h)$ centrado en x

$$u(x+h) = u(x) + u'(x)(\cancel{x}+h-\cancel{x}) + \frac{u''(c)}{2}(\cancel{x}+h-\cancel{x})^2$$

$$= u(x) + u'(x)h + \frac{u''(c)\,h^2}{2}$$

Reemplazamos en

$$E = \left| u'(x) - \underbrace{\frac{u(x+h) - u(x)}{h}}_{D_+u(x)} \right|$$

$$E = \left| \cancel{u'(x)} - \left(\frac{\left(\cancel{u(x)} + \cancel{u'(x)}\cancel{h} + \frac{u''(x)h^{2}}{2}\right) - \cancel{u(x)}}{\cancel{h}} \right) \right|$$

$$E = \left| \frac{u''(x)\,h}{2} \right| = \frac{|u''(x)|\,h}{2} \rightarrow O(h)$$

b) $$E = \left| u'(x) - \left(\frac{u(x) - u(x-h)}{h} \right) \right|$$

Taylor de $u(x-h)$ en x

$$P(x) = u(x) + u'(x)(\cancel{x} - h - \cancel{x}) + \frac{u''(x)}{2}(\cancel{x} - h - \cancel{x})^{2}$$
$$= u(x) - u'(x)h + \frac{u''(x)}{2}h^{2}$$

Reemplazo

$$E = \left| \cancel{u'(x)} - \left(\frac{\cancel{u(x)} - \left(\cancel{u(x)} - \cancel{u'(x)}\cancel{h} + \frac{u''(x)}{2}h^{\cancel{2}}\right)}{\cancel{h}} \right) \right|$$

$$E = \left| \frac{u''(x)\,h}{2} \right| = \frac{|u''(x)|\,h}{2} \rightarrow O(h)$$

c)

$$E = \left| U'(x) - \frac{U(x+h) - U(x-h)}{2h} \right|$$

Taylor grado 2

$$U(x+h) = U(x) + U'(x)h + U''(x)\frac{h^2}{2} + U'''(x)\frac{h^3}{6}$$

$$U(x-h) = U(x) - U'(x)h + U''(x)\frac{h^2}{2} - U'''(x)\frac{h^3}{6}$$

Volviendo a reemplazar

$$E = \left| U'(x) - \left(\frac{\left.\begin{array}{l} \cancel{U(x)} + U'(x)\cancel{h} + \cancel{U''(x)\frac{h^2}{2}} + U'''(c_1)\frac{h^{\cancel{3}\,2}}{6} \\ - \left(\cancel{U(x)} - U'(x)\cancel{h} + \cancel{U''(x)\frac{h^2}{2}} - U'''(c_2)\frac{h^{\cancel{3}\,2}}{6}\right) \end{array}\right] \begin{array}{l} U(x+h) \\ U(x-h) \end{array}}{2h} \right) \right|$$

$$E = \left| \cancel{U'(x)} - \left(\frac{\cancel{2}\cancel{U'(x)} + U'''(c_1)\frac{h^2}{3}}{\cancel{2}} \right) \right|$$

$$E = \left| \frac{U'''(x)h^2}{6} \right| = \frac{|U'''(c_2)|\,h^2}{6} \rightarrow O(h^2)$$

d)

Que no me cruce al que hizo este ej porque lo mato

Taylor grado 3 de todo:

En verde calculo la cuenta del corchete

$$2\left(u(x+h) = \underset{2}{\cancel{u(x)}} + \underset{2}{u'(x)\,h} + \underset{1}{\cancel{\frac{u''(x)h^2}{2}}} + \underset{\frac{1}{3}}{\cancel{\frac{u'''(x)h^3}{6}}} + \frac{u^{iv}(c_1)\,h^4}{24}\right)$$

$+$

$$3\left(U(x) = \underset{5}{\cancel{u(x)}}\right)$$

$-$

$$6\left(u(x-h) = \underset{-1}{\cancel{u(x)}} - \underset{8}{u'(x)\,h} + \underset{-2}{\cancel{\frac{u''(x)h^2}{2}}} - \underset{\frac{4}{3}}{\cancel{\frac{u'''(x)h^3}{6}}} + \frac{u^{iv}(c_2)\,h^4}{24}\right)$$

$+$

$$\left(u(x-2h) = \underset{0}{\cancel{u(x)}} - \underset{6}{u'(x)\,2h} + \underset{0}{\cancel{\frac{u''(x)4h^2}{2}}} - \underset{0}{\cancel{\frac{u'''(x)8h^3}{6}}} + \frac{u^{iv}(c_3)16h^4}{24}\right)$$

Reescribo

$$D_3 u(x) = \frac{1}{6h}\left(u'(x)\,6h + \frac{\left(u^{iv}(c_1) + u^{iv}(c_2) + 16u^{iv}(c_3)\right)h^4}{24}\right)$$

$$E = \left|\cancel{u'(x)} - \cancel{\frac{1}{6h}}\left(\cancel{u'(x)6h} + \frac{\left(u^{iv}(c_1) + u^{iv}(c_2) + 16u^{iv}(c_3)\right)h^{\cancel{4}\,3}}{24}\right)\right|$$

$$E = \frac{\left|u^{iv}(c_1) + u^{iv}(c_2) + 16u^{iv}(c_3)\right|}{24}h^3 \rightarrow O(h^3)$$

1.6 Aproxime la derivada de alguna (a su elección) función f utilizando la siguiente fórmula

$$f_n'(x) = \frac{f(x + 10^{-n}) - f(x)}{10^{-n}}$$

para distintos valores de $n = 0, 1, 2, \cdots, 15$. Compare con el valor exacto y analice los resultados.

```python
def f(x):
    return np.arctan(x)

def df(x):
    return 1 / (1 + x**2)

def formula(f, x, n):
    return (f(x + 10**-n) - f(x)) / (10**-n)

for n in range(16):
    for x in range(-1, 2):
        print(f"n = {n}: f'({x}) = {formula(f, x, n)} - error = {abs(formula(f, x, n) - df(x))}")
    print("------------------------------------------------------------------")
```

```
n = 0: f'(-1) = 0.7853981633974483 - error = 0.2853981633974483
n = 0: f'(0) = 0.7853981633974483 - error = 0.21460183660255172
n = 0: f'(1) = 0.32175055439664213 - error = 0.17824944560335787
------------------------------------------------------------------
n = 1: f'(-1) = 0.5258306161094173 - error = 0.02583061610941728
n = 1: f'(0) = 0.9966865249116204 - error = 0.003313475088379647
n = 1: f'(1) = 0.4758310327698345 - error = 0.024168967230165483
------------------------------------------------------------------
n = 2: f'(-1) = 0.5025083330812419 - error = 0.002508333081241876
n = 2: f'(0) = 0.9999666686665237 - error = 3.333133347627193e-05
n = 2: f'(1) = 0.49750833308541687 - error = 0.002491666914583135
------------------------------------------------------------------
n = 3: f'(-1) = 0.5002500833333201 - error = 0.0002500833333201058
n = 3: f'(0) = 0.9999996666668668 - error = 3.333331320671465e-07
n = 3: f'(1) = 0.4997500833332502 - error = 0.0002499166667497832
------------------------------------------------------------------
n = 4: f'(-1) = 0.5000250008324603 - error = 2.5000832460264633e-05
n = 4: f'(0) = 0.9999999966666666 - error = 3.333333387089965e-09
n = 4: f'(1) = 0.49997500083387436 - error = 2.4999166125638794e-05
------------------------------------------------------------------
n = 5: f'(-1) = 0.5000025000034825 - error = 2.5000034824529394e-06
n = 5: f'(0) = 0.9999999999666667 - error = 3.33333360913457e-11
n = 5: f'(1) = 0.4999975000141709 - error = 2.4999858290741805e-06
------------------------------------------------------------------
n = 6: f'(-1) = 0.5000002499810918 - error = 2.499810918266121e-07
...
n = 15: f'(-1) = 0.4440892098500626 - error = 0.05591079014993744
n = 15: f'(0) = 1.0 - error = 0.0
n = 15: f'(1) = 0.5551115123125783 - error = 0.05511151231257827
```

dejo algunos nomas porque no entra en la terminal

1.7 Dada la expresión para el error

$$e(h) = \frac{\epsilon}{h} + \frac{h^2}{6}M,$$

donde M es una cota para la derivada tercera de una función. Muestre que $e(h)$ tiene un mínimo en $\sqrt[3]{3\epsilon/M}$.

$M \to$ cota , $\underline{f'''(c) < M}$

Buscamos $e'(h)$ tal que $e'\left(\sqrt{\frac{3\epsilon}{M}}\right) = 0$

$$e'(h) = -\frac{\epsilon}{h^2} + \frac{h}{3}M = 0$$

$$\frac{h}{3}M = \frac{\epsilon}{h^2}$$

$$h^3 = \frac{3\epsilon}{M}$$

$$\Rightarrow \boxed{h = \sqrt{\frac{3\epsilon}{M}}}$$

$\to$ tiene extremo en $h = \sqrt[3]{\frac{3\epsilon}{M}}$

Chequeamos que sea mínimo

$$e''(h) = \frac{M}{3} + \frac{2\epsilon}{h^3}$$

$$\Rightarrow e''\left(\sqrt{\frac{3\epsilon}{M}}\right) = \frac{M}{3} + \frac{2(\epsilon)}{\left(\sqrt[\cancel{3}]{\frac{3\epsilon}{M}}\right)^{\cancel{3}}} = \frac{M}{3} + \frac{2\cancel{\epsilon}}{3\cancel{\epsilon}}M$$

$$\Rightarrow e''\left(\sqrt{\frac{3\epsilon}{M}}\right) = M \to M \oplus \ ?$$

1.8 Sea la función

$$f(x, y) = x^2 e^y + \sin(xy).$$

Aproximar la derivada parcial respecto de x, $\frac{\partial f}{\partial x}$, y de y, $\frac{\partial f}{\partial y}$ en el punto $(x, y) = (1, 0.5)$. Utilizar para cada caso diferencias adelantadas y diferencias centradas de 2 puntos tomando $h = 0.01$.

Comparar los resultados de cada esquema con el valor exacto de las derivadas parciales.

$$\frac{\partial f}{\partial x} = 2x\,e^y + \cos(xy)\,y$$

$$\frac{\partial f}{\partial y} = x^2 e^y + \cos(xy)\,x$$

```
h = 0.01
x, y = (1.0, 0.5)

def f(x, y):
    return x**2 * np.exp(y) + np.sin(x * y)

def dfdx(x, y):
    return 2 * x * np.exp(y) + y * np.cos(x * y)

def dfdy(x, y):
    return x**2 * np.exp(y) + x * np.cos(x * y)

def derivadaAprox(f, xn0, xn1):
    return (f(xn1) - f(xn0)) / (xn1 - xn0)

print(f"df/dx = {dfdx(x, y)}")
print(f"df/dx aprox adelantada = {derivadaAprox(lambda x: f(x, y), x, x + h)}")
print(f"df/dx aprox centrada = {derivadaAprox(lambda x: f(x, y), x - h, x + h)}")
print()
print(f"df/dy = {dfdy(x, y)}")
print(f"df/dy aprox adelantada = {derivadaAprox(lambda y: f(x, y), y, y + h)}")
print(f"df/dy aprox centrada = {derivadaAprox(lambda y: f(x, y), y - h, y + h)}")
```

```
df/dx = 3.7362338223454428
df/dx aprox adelantada = 3.752119926082994
df/dx aprox centrada = 3.7362319940507285

df/dy = 2.526303832590501
df/dy aprox adelantada = 2.532163252446249
df/dy aprox centrada = 2.5263166851128482
```