

- 1) Un bloque de masa m desliza bajo la acción de la gravedad por un plano inclinado formando un ángulo θ respecto de la horizontal. Se puede demostrar que, si la fuerza de rozamiento F_r entre el bloque y el plano viene dada por $F_r = -kmv^2$ con v la velocidad del bloque y k un coeficiente de rozamiento, entonces el tiempo T requerido para que el bloque recorra una distancia D partiendo del reposo está relacionado de la forma:

$$e^{kD} = \cosh(T\sqrt{kg\sin\theta})$$

siendo $g = 9,8$ la aceleración de la gravedad. Si $\theta = \frac{\pi}{4}$, $k = 0,5 \pm 0,1$ y $T = 2 \pm 0,2$, hallar la precisión con la que se conoce D .

Para hallar la precisión con la que se conoce D , debemos propagar las incertidumbres de las variables k y T a través de la ecuación dada:

$$e^{kD} = \cosh(T\sqrt{kg\sin\theta})$$

Primero, despejamos D :

$$kD = \ln[\cosh(T\sqrt{kg\sin\theta})] \Rightarrow D = \frac{1}{k} \ln[\cosh(T\sqrt{kg\sin\theta})]$$

Calculamos D usando los valores centrales:

- $\theta = \frac{\pi}{4} \Rightarrow \sin\theta = \frac{\sqrt{2}}{2} \approx 0.7071$
- $k = 0.5$
- $T = 2$
- $g = 9.8 \text{ m/s}^2$

Calculamos $A = T\sqrt{kg\sin\theta}$:

$$A = 2 \times \sqrt{0.5 \times 9.8 \times 0.7071} \approx 2 \times 1.8612 \approx 3.7224$$

Luego, calculamos D :

$$D = \frac{1}{0.5} \ln[\cosh(3.7224)] \approx 2 \times \ln(20.6896) \approx 6.060 \text{ metros}$$

Ahora, calculamos las derivadas parciales de D respecto a k y T :

1. **Derivada respecto a k :**

$$\frac{\partial D}{\partial k} = -\frac{1}{k^2} \ln[\cosh(A)] + \frac{1}{k} \tanh(A) \left(\frac{\partial A}{\partial k} \right)$$

Donde:

$$\frac{\partial A}{\partial k} = T \times \frac{g \sin\theta}{2\sqrt{kg\sin\theta}} = \frac{Tg \sin\theta}{2\sqrt{kg\sin\theta}}$$

Calculamos los valores numéricos:

$$\frac{\partial A}{\partial k} \approx \frac{2 \times 9.8 \times 0.7071}{2 \times 1.8612} \approx 3.7261$$

Entonces:

$$\frac{\partial D}{\partial k} \approx -\frac{1}{(0.5)^2} \times 3.03 + \frac{1}{0.5} \times 0.9988 \times 3.7261 \approx -12.12 + 7.442 \approx -4.678$$

2. **Derivada respecto a T :**

$$\frac{\partial D}{\partial T} = \frac{1}{k} \tanh(A) \left(\frac{\partial A}{\partial T} \right)$$

Donde:

$$\frac{\partial A}{\partial T} = \sqrt{kg\sin\theta} \approx 1.8612$$

Calculamos:

$$\frac{\partial D}{\partial T} \approx \frac{1}{0.5} \times 0.9988 \times 1.8612 \approx 3.718$$

Finalmente, propagamos las incertidumbres:

$$\Delta D = \sqrt{\left(\frac{\partial D}{\partial k} \Delta k\right)^2 + \left(\frac{\partial D}{\partial T} \Delta T\right)^2}$$

Con $\Delta k = 0.1$ y $\Delta T = 0.2$:

$$\Delta D = \sqrt{(-4.678 \times 0.1)^2 + (3.718 \times 0.2)^2} \approx \sqrt{0.2188 + 0.5530} \approx 0.878 \text{ metros}$$

Respuesta final:

La precisión con la que se conoce D es aproximadamente $\Delta D = \pm 0.88$ metros.

Por lo tanto, $D = 6.06 \pm 0.88$ metros.

✓ 2) Utilizando el método de redondeo.

✓ a) Hallar el número de máquina más próximo a 125, 6 y a 126 si trabaja con

Base 10 y mantisa de 2 dígitos:

En primera instancia, dado que vamos a trabajar con base 10, reescribimos los números en notación científica de tal forma que la parte entera sea 0. Los números quedan de la siguiente forma:

$$125.6 = 1.256 \times 10^{-2}$$

y

$$126 = 1.26 \times 10^{-2}$$

Ahora, recordemos que la mantisa representa las cifras decimales del número. En nuestro caso, la mantisa tiene dos dígitos, mientras que los números con los que trabajamos tienen cuatro y tres dígitos respectivamente, por lo que realizamos un redondeo.

Para el primer número, 1.256×10^{-2} , notemos que el tercer dígito de la mantisa es 5. Según el método de redondeo, como este es igual a $\frac{\beta}{2}$, donde $\beta = 10$ es la base, el tercer dígito se incrementa en un valor. Así, en base 10 y con mantisa de dos dígitos, el número 125.6 se redondea a:

$$1.256 \times 10^{-2} = 1.3 \times 10^{-2}$$

Repitiendo este proceso con 1.26×10^{-2} , y dado que $6 \geq \frac{\beta}{2}$, el número se redondea a:

$$1.26 \times 10^{-2} = 1.3 \times 10^{-2}$$

Por consiguiente, el número de máquina más cercano a 125.6 y 126 es 130.

Base 2 y mantisa de 8 dígitos

De forma similar a lo anterior, reescribimos los números en base 2. Esto se realiza separando la parte entera de la decimal y convirtiéndolas por separado, como se muestra a continuación:

Para 125:

Se divide sucesivamente entre 2, registrando los residuos:

División	Residuo
$125 \div 2 = 62$	Residuo: 1
$62 \div 2 = 31$	Residuo: 0
$31 \div 2 = 15$	Residuo: 1
$15 \div 2 = 7$	Residuo: 1
$7 \div 2 = 3$	Residuo: 1
$3 \div 2 = 1$	Residuo: 1
$1 \div 2 = 0$	Residuo: 1

Leyendo los residuos de abajo hacia arriba:

$$125 = 1111101$$

Para 126:

El proceso es similar, pero como 126 es par, el primer residuo será 0:

División	Residuo
$126 \div 2 = 63$	Residuo: 0
$63 \div 2 = 31$	Residuo: 1
$31 \div 2 = 15$	Residuo: 1
$15 \div 2 = 7$	Residuo: 1
$7 \div 2 = 3$	Residuo: 1
$3 \div 2 = 1$	Residuo: 1
$1 \div 2 = 0$	Residuo: 1

Leyendo los residuos de abajo hacia arriba:

$$126 = 1111110$$

Parte decimal: 0.6

Se multiplica sucesivamente por 2, registrando las partes enteras:

Multiplicación	Parte entera
$0.6 \times 2 = 1.2$	Parte entera: 1
$0.2 \times 2 = 0.4$	Parte entera: 0
$0.4 \times 2 = 0.8$	Parte entera: 0
$0.8 \times 2 = 1.6$	Parte entera: 1
$0.6 \times 2 = 1.2$	Parte entera: 1

Esto genera la secuencia 0.10011, observando que es periódica.

Por consiguiente, los números quedan reescritos de la siguiente forma:

- Para 125.6:

$$125.6 = 111101.10011 = 1.1110110011 \times 2^6$$

- Para 126:

$$126 = 1111110 = 1.111110 \times 2^6$$

Una vez descrito de esta forma y dado que nuestra mantisa tiene únicamente 8 dígitos después del punto, realizamos redondeo. Para el primer caso, 1.1110110011×2^6 , como el noveno dígito decimal es 0 y $0 < \frac{\beta}{2} = 1$, realizamos un truncamiento en el octavo dígito decimal. Así, obtenemos:

$$125.6 \approx 1.1110110 \times 2^6$$

Por ende, el número de máquina más cercano es:

$$1.1110110 \times 2^6 = \left(1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{16} + 0 + \frac{1}{64} + \frac{1}{128}\right) \times 2^6 = 125.5$$

En el caso de 126, como su expresión en binario tiene menos de 8 cifras decimales, su representación nos queda igual. Es decir, el número de máquina más cercano a 125.6 es 125.5 y a 126 es 126.

✓ b) Verificar para $x = 125,6$ la cota para el error relativo

$$\left| \frac{x - fl(x)}{x} \right| \leq \epsilon$$

si $\epsilon = \frac{1}{2}\beta^{1-d}$ donde β es la base y d la longitud de la mantisa.

Base 10 y mantisa de 2 dígitos:

Según la teoría, sabemos que para $\beta = 10$ y $d = 2$, la cota para el error relativo es:

$$\epsilon_1 = \frac{1}{2} \cdot 10^{1-2} = 5.0 \times 10^{-2}$$

Verifiquemos que esto se cumple con los datos que encontramos:

$$\left| \frac{(125.6) - fl(125.6)}{125.6} \right| = \left| \frac{125.6 - 130}{125.6} \right| \approx 3.5 \times 10^{-2} < \epsilon_1$$

Base 2 y mantisa de 8 dígitos:

Ahora, tomando $\beta = 2$ y $d = 9$ pues ganamos un dígito con la normalización, tenemos que:

$$\epsilon_2 = \frac{1}{2} \cdot 2^{1-9} = 2^{-9} \approx 1.95 \times 10^{-3}$$

Verifiquemos que esta cota se mantiene para los datos encontrados:

$$\left| \frac{(125.6) - fl(125.6)}{125.6} \right| = \left| \frac{125.6 - 125.5}{125.6} \right| \approx 7.96 \times 10^{-4} < \epsilon_2$$

En conclusión, los errores relativos se mantiene por debajo de las cotas esperadas.

- ✓ c) ¿Cuál es, en cada caso, el valor que da la máquina como resultado de las operaciones $126 + 126.5$ y $126 - 125.6$? ¿Cuál es el error relativo de estos resultados?

Base 10 y mantisa de 2 dígitos

Primero, como no conocemos el número de máquina más cercano a 126.5, procederemos a calcularlo. Escribimos el número en base 10 y redondeamos a dos dígitos significativos, obteniendo:

$$126.5 = 1.265 \times 10^2 \approx 1.30 \times 10^2$$

Con las representaciones de máquina de todos los números, procedemos a calcular los valores de máquina de $126 + 126.5$ y $126 - 125.6$:

1. **Para $126 + 126.5$:**

$$fl(126 + 126.5) = fl(fl(126) + fl(126.5)) = fl(130 + 130) = fl(260) = 2.60 \times 10^2 = 260$$

2. **Para $126 - 125.6$:**

$$fl(126 - 125.6) = fl(fl(126) - fl(125.6)) = fl(130 - 130) = fl(0) = 0$$

Ahora, calculamos los errores relativos de estos resultados:

1. **Error relativo de $126 + 126.5$:**

$$\left| \frac{fl(126 + 126.5) - (126 + 126.5)}{126 + 126.5} \right| = \left| \frac{260 - 252.5}{252.5} \right| \approx 2.93 \times 10^{-2}$$

2. **Error relativo de $126 - 125.6$:**

$$\left| \frac{fl(126 - 125.6) - (126 - 125.6)}{126 - 125.6} \right| = \left| \frac{0 - 0.4}{0.4} \right| = 1$$

En resumen, los valores de máquina para las operaciones son 260 y 0, y los errores relativos respectivos son aproximadamente 2.93×10^{-2} y 1.

Base 2 y mantisa de 8 dígitos

Nuevamente, como no conocemos el número de máquina más cercano a 126.5, procederemos a calcularlo. Para convertir 126.5 a binario con una mantisa de 8 dígitos, primero convertimos la parte entera 126 dividiendo sucesivamente por 2, lo que da $126 = 1111110$. Luego, la parte decimal 0.5 se convierte multiplicando por 2, obteniendo $0.5 = 0.1$. Al combinar ambas partes, 126.5 en binario es 1111110.1.

Posteriormente, normalizamos el número, expresándolo como $1111110.1 = 1.1111101 \times 2^6$, donde la parte decimal tiene 7 dígitos. Así, la representación binaria final es $126.5 = 1.1111101 \times 2^6$.

Con las representaciones de máquina de todos los números, procedemos a calcular los valores de máquina de $126 + 126.5$ y $126 - 125.6$:

1. **Para $126 + 126.5$:**

$$fl(126 + 126.5) = fl(fl(126) + fl(126.5)) = fl(126 + 126.5) = fl(252.5) = fl(1.11111001 \times 2^7) = 1.11111001 \times 2^7 = 252.5$$

2. **Para $126 - 125.6$:**

$$fl(126 - 125.6) = fl(fl(126) - fl(125.6)) = fl(126 - 125.5) = fl(0.5) = fl(1.00000000 \times 2^{-1}) = 1.00000000 \times 2^{-1} = 0.5$$

Ahora, calculamos los errores relativos de estos resultados:

1. **Error relativo de $126 + 126.5$:**

$$\left| \frac{fl(126 + 126.5) - (126 + 126.5)}{126 + 126.5} \right| = \left| \frac{252.5 - 252.5}{252.5} \right| = 0$$

2. **Error relativo de $126 - 125.6$:**

$$\left| \frac{fl(126 - 125.6) - (126 - 125.6)}{126 - 125.6} \right| = \left| \frac{0.5 - 0.4}{0.4} \right| = 0.25 = 2.5 \times 10^{-1}$$

En resumen, los valores de máquina para las operaciones son 252.5 y 0.5, y los errores relativos respectivos son 0 y 2.5×10^{-1} .

- 3) Suponga que una compañía de computadores esta desarrollando un nuevo sistema de punto flotante para usarlo con sus máquinas. Ellos necesitan su ayuda para responder unas preguntas acerca de su sistema. Siguiendo la teminología descrita en clase, el sistema de punto flotante de la compañía se especifica por (β, t, L, U) . Usted debe suponer que:

- Todos los valores de punto flotante son normalizados (excepto la representación de punto flotante de cero).
- Todos los dígitos en la mantisa de un valor de punto flotante son almacenados explícitamente.
- El cero se representa con una mantisa y exponente de ceros.

Preguntas

- a) ¿ Cuántos valores diferentes de punto flotante no negativos pueden representarse por medio de este sistema de punto flotante ?

Como sabemos el sistema de punto flotante que nos entrega la empresa (β, t, L, U) , donde:

- β = base del sistema numérico.
- t = precisión (número de dígitos).
- L = cota inferior del exponente e .
- U = cota superior del exponente e .

donde:

$$fl(x) = \pm \left(d_0 + \frac{d_1}{\beta} + \frac{d_2}{\beta^2} + \dots + \frac{d_{t-1}}{\beta^{t-1}} \right) \beta^e = \pm (d_0.d_1d_2 \dots d_{t-1})\beta^e$$

donde los d_i son dígitos en el rango $0 \leq d_i \leq \beta - 1$ y se obliga a que $d_0 \neq 0$ a menos de que se esté representando el 0 (esto es ser un sistema de punto flotante normalizado).

Siendo así, como los d_i con $i \neq 0$ varían de 0 a $\beta - 1$ sabemos que cada uno puede tomar β opciones, y como d_0 toma valores entre 1 y $\beta - 1$, esta puede tomar $(\beta - 1)\beta^{t-1}$ combinaciones distintas.

Ahora, si pensamos en el exponente e , este puede variar entre L y U , es decir $L \leq e \leq U$, lo que implica que e puede tomar $U - L + 1$ valores distintos, por lo que podemos concluir que la cantidad de valores distintos de punto flotante no negativos que pueden representarse por medio de este sistema de punto flotante son:

$$(\beta - 1)\beta^{t-1}(U - L + 1)$$

- b) La misma pregunta para el caso $(\beta, t, L, U) = (8, 5, -100, 100)$, el cual la compañía esta contemplando en particular.

Utilizando el razonamiento anterior y tomando $(\beta, t, L, U) = (8, 5, -100, 100)$ llegamos a:

$$\begin{aligned} (\beta - 1)\beta^{t-1}(U - L + 1) &= (8 - 1)(8)^{5-1}(100 - (-100) + 1) \\ &= (7)(4096)(201) \\ &= 5763072 \end{aligned}$$

lo que nos permite concluir que la cantidad de valores distintos de punto flotante no negativos que pueden representarse por medio de este sistema de punto flotante $(8, 5, -100, 100)$ son 5763.072.

- c) ¿Cuál es el valor aproximado (en base 10) del número más grande y el número positivo más pequeño que pueden ser representado en este sistema de punto flotante?

Sabemos que el valor más grande que puede tomar la mantisa en base 8 es cuando todos los d_i toman el valor $\beta - 1 = 7$, por otro lado también sabemos que el exponente debe ser el mayor, es decir, $U = 100$.

Por lo tanto el número más grande que puede ser representado en este sistema de punto flotante es:

$$\begin{aligned} (7.7777)(8)^{100} &= \left(7 + \frac{7}{8} + \frac{7}{8^2} + \frac{7}{8^3} + \frac{7}{8^4} \right) (8)^{100} \\ &= (7 * 8^4 + 7 * 8^3 + 7 * 8^2 + 7 * 8 + 7)(8)^{96} \\ &= (7)(8)^{96}(8^4 + 8^3 + 8^2 + 8 + 1) \\ &= (7)(8)^{96}(4681) \\ &= 16295790487439478903505410125026878467571646693176692657371683130858157238609681116091645952 \\ &\approx 1.629 \times 10^{52} \end{aligned}$$

Por otro lado, para conocer el número positivo más pequeño que puede ser representado en este sistema de punto flotante sabemos que d_0 debe ser igual a 1 y que d_i con $1 \leq i \leq t-1$ deben de ser 0, de forma similar al punto anterior sabemos que el exponente debe de ser el menor, es decir, $L = -100$. Siendo así este número es:

$$(1.0000)(8)^{-100} \approx 7.8886 \times 10^{-91}$$

✓ d) Para el caso general, demuestre que los errores relativos producidos por realizar truncamiento y redondeo son

$$\left| \frac{fl(x) - x}{x} \right| = \begin{cases} \beta^{1-t} & \text{Cuando se efectua el truncamiento.} \\ \frac{\beta^{1-t}}{2} & \text{Cuando se efectua el redondeo.} \end{cases}$$

Primero veamos el caso cuando se efectua el redondeo.

Suponga $x > 0$ con $d_t > \frac{\beta}{2}$ y calculamos $fl(x) - x$, entonces:

$$\begin{aligned} fl(x) - x &= (d_0, d_1 d_2 \cdots d_{t-1} + 1 \quad 0000 \cdots) \beta^e - (d_0, d_1 d_2 \cdots d_{t-1} \quad d_t d_{t+1} d_{t+2} d_{t+3} \cdots) \beta^e \\ &= (0, 00 \cdots 0 \quad \beta - 1 - d_t \quad \beta - 1 - d_{t+1} \quad \beta - 1 - d_{t+2} \quad \beta - 1 - d_{t+3} \cdots) \beta^e \\ &= (\beta - 1 - d_t, \beta - 1 - d_{t+1} \quad \beta - 1 - d_{t+2} \quad \beta - 1 - d_{t+3} \quad \cdots) \beta^{e-t} \\ &\leq \frac{1}{2} \beta \beta^{e-t} \end{aligned}$$

la desigualdad anterior se da, ya que $d_t > \frac{\beta}{2}$ y por lo tanto $\beta - 1 - d_t \leq \beta - 1 - \frac{\beta}{2} \leq \frac{\beta}{2}$. Por otra parte tenemos que como $d_0 > 0$, entonces $|x| \geq \beta^e$, de lo que se deduce que:

$$\begin{aligned} \frac{|fl(x) - x|}{|x|} &\leq \frac{\frac{1}{2} \beta \beta^{e-t}}{\beta^e} \\ &\leq \frac{1}{2} \beta^{1-t} \end{aligned}$$

Ahora veamos el caso de truncamiento, para esto calculemos de forma análoga al paso anterior $fl(x) - x$.

$$\begin{aligned} fl(x) - x &= (d_0, d_1 d_2 \cdots d_{t-1} \quad 000 \cdots) \beta^e - (d_0, d_1 d_2 \cdots d_{t-1} \quad d_t d_{t+1} d_{t+2} \cdots) \beta^e \\ &= (0, 00 \cdots 0 \quad \beta - 1 - d_t \quad \beta - 1 - d_{t+1} \quad \beta - 1 - d_{t+2} \quad \cdots) \beta^e \\ &= (\beta - 1 - d_t, \beta - 1 - d_{t+1} \quad \beta - 1 - d_{t+2} \quad \cdots) \beta^{e-t} \\ &\leq \beta \beta^{e-t} \end{aligned}$$

Y de igual forma al caso anterior $|x| \geq \beta^e$, ya que $d_0 > 0$, entonces:

$$\begin{aligned} \frac{|fl(x) - x|}{|x|} &\leq \frac{\beta \beta^{e-t}}{\beta^e} \\ &= \beta^{1-t} \end{aligned}$$

Con lo que podríamos concluir que:

$$\left| \frac{fl(x) - x}{x} \right| = \begin{cases} \beta^{1-t} & \text{Cuando se efectua el truncamiento.} \\ \frac{\beta^{1-t}}{2} & \text{Cuando se efectua el redondeo.} \end{cases}$$

✓ 4) Considere la expansión de Taylor para la función exponencial

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots = \sum_{i=0}^{\infty} \frac{x^i}{i!} = \lim_{N \rightarrow \infty} S(x, N)$$

donde $S(x, N)$ es la suma parcial con $N + 1$ términos.

✓ Preguntas

- a) Escriba un programa que grafique el error relativo de la suma, $\frac{|S(x, N) - e^x|}{e^x}$ versus N (hasta $N = 60$) para un valor dado de x . Pruebe su programa para $x = 10, 2, -2$ y -10 . De las gráficas, explique por qué ésta no es una buena manera para evaluar e^x cuando $x < 0$.

Queremos graficar el error relativo de esta suma parcial respecto a e^x , definido como:

$$\text{Error relativo} = \frac{|S(x, N) - e^x|}{e^x}$$

Esto nos ayudará a ver qué tan buena es la aproximación cuando tomamos distintos valores de N .

Para esto, usaremos un programa hecho en Python, que nos ayudará a ver la gráfica del error.

Primero vamos a importar las librerías que nos ayudarán a realizar cálculos y graficas matemáticas, esto es:

```
import numpy as np
import matplotlib.pyplot as plt
```

Ahora, definiremos la función S como la suma parcial, esto es:

```
def S(x, N):
    suma = 0
    for i in range(N + 1): # i va de 0 a N
        suma += x**i / np.math.factorial(i)
    return suma
```

Luego vamos a crear una función que nos va a ayudar a calcular el error relativo:

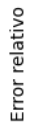
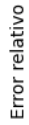
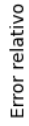
```
def graficar_error_relativo(x):
    errores_relativos = []
    valores_N = range(61) # N va de 0 a 60
    ex = np.exp(x) # El valor exacto de e^x usando numpy

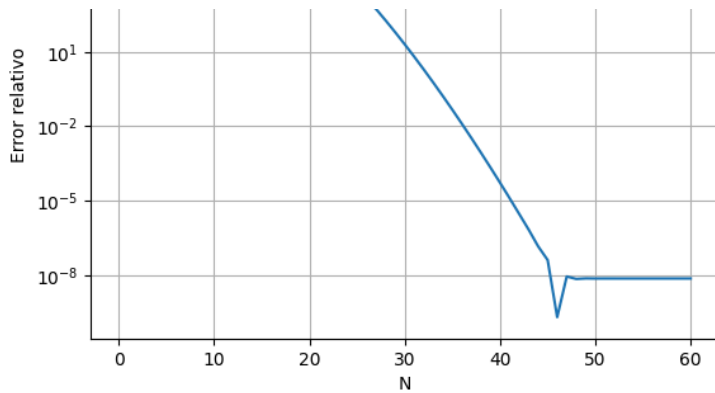
    for N in valores_N:
        SxN = S(x, N)
        error_relativo = abs(SxN - ex) / ex
        errores_relativos.append(error_relativo)

    plt.plot(valores_N, errores_relativos, label=f"x = {x}")
    plt.yscale("log") # Escala logarítmica para ver bien el error
    plt.xlabel("N")
    plt.ylabel("Error relativo")
    plt.title(f"Error relativo de S(x, N) vs N para x = {x}")
    plt.legend()
    plt.grid()
    plt.show()
```

Ahora pondremos en uso nuestro programa con los valores asignados por el problema (10, 2, -2 y -10), esto es:

```
for x in [10, 2, -2, -10]:
    graficar_error_relativo(x)
```





- ✓ b) Modifique su programa tal que use la identidad $e^x = \frac{1}{e^{-x}} = \frac{1}{S(-x, \infty)}$ para evaluar la función exponencial cuando x es negativa. Explique por qué esta técnica funciona mejor.

Ahora, note que para valores de x negativos, la expresión e^x se vuelve muy pequeña, por lo cuál nos será de gran ayuda usar $e^x = \frac{1}{e^{-x}}$, pues e^{-x} probablemente no sea tan pequeño, si no por el contrario un valor grande el cuál evaluar en $\frac{1}{e^{-x}}$.

Para esto modificaremos nuestro código de cálculo para S y por ende la función de graficar el error relativo:

```
def S_modificado(x, N):
    if x >= 0:
        return S(x, N)
    else:
        return 1 / S(-x, N)

def graficar_error_relativo_modificado(x):
    errores_relativos = []
    valores_N = range(61) # N va de 0 a 60
    ex = np.exp(x) # El valor exacto de e^x usando numpy

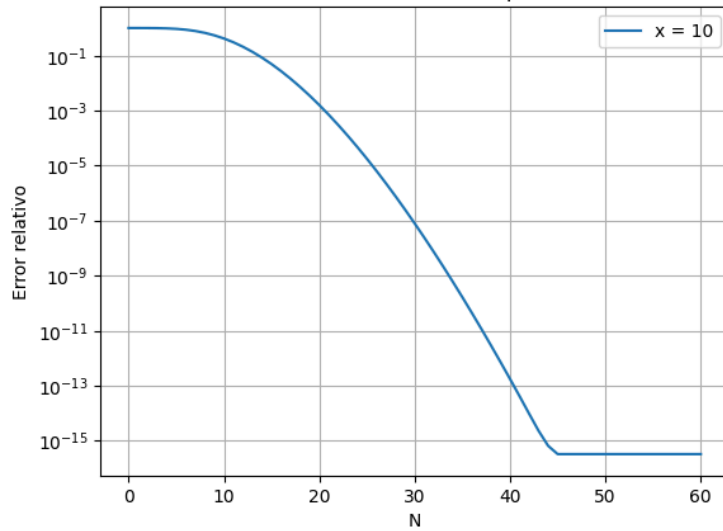
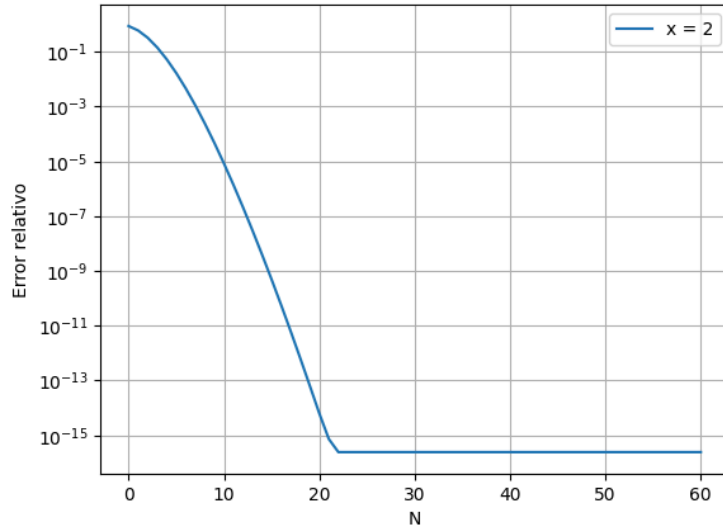
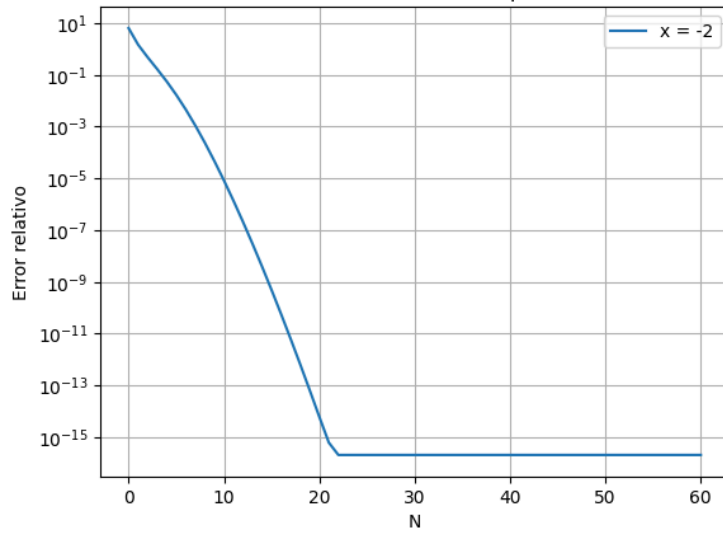
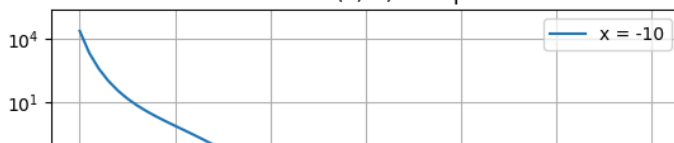
    for N in valores_N:
        SxN = S_modificado(x, N)
        error_relativo = abs(SxN - ex) / ex
        errores_relativos.append(error_relativo)

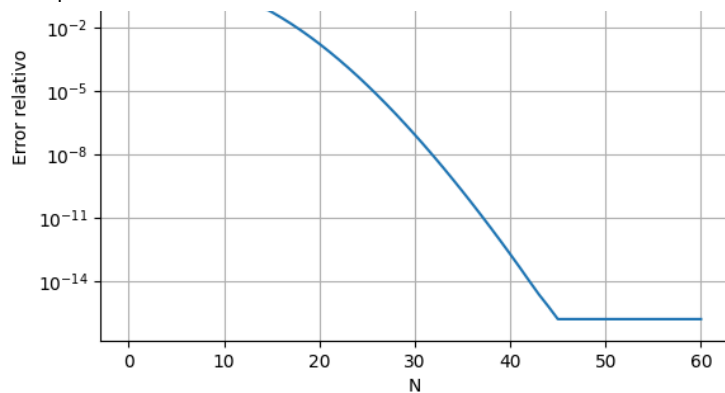
    plt.plot(valores_N, errores_relativos, label=f"x = {x}")
    plt.yscale("log") # Escala logarítmica para ver bien el error
    plt.xlabel("N")
    plt.ylabel("Error relativo")
    plt.title(f"Error relativo de S(x, N) vs N para x = {x}")
    plt.legend()
    plt.grid()
    plt.show()
```

Ahora pondremos en uso nuestro programa con los valores asignados por el problema (10, 2, -2 y -10), esto es:

```
for x in [10, 2, -2, -10]:
    graficar_error_relativo_modificado(x)
```

```
<ipython-input-2-4a069650a4db>:4: DeprecationWarning: `np.math` is a deprecated alias for the standard library `math` module (Dep  
suma += x**i / np.math.factorial(i)
```

Error relativo de $S(x, N)$ vs N para $x = 10$ Error relativo de $S(x, N)$ vs N para $x = 2$ Error relativo de $S(x, N)$ vs N para $x = -2$ Error relativo de $S(x, N)$ vs N para $x = -10$ 



✓ 5)

✓ a) De manera similar a la desarrollada en clase, deduzca que una aproximación de $f'(x_0)$ es

$$\frac{f(x_0 + h) - f(x_0 - h)}{2h}.$$

Muestre que esta aproximación tiene un error de $O(h^2)$. Más precisamente, el primer término del error es $-\frac{h^2}{6} f'''(x_0)$ cuando $f'''(x_0) \neq 0$.

De manera similar a lo desarrollado en clase, deduzca que una aproximación de $f'(x_0)$ es

$$\frac{f(x_0 + h) - f(x_0 - h)}{2h}.$$

Muestre que esta aproximación tiene un error de $O(h^2)$. Más precisamente, el primer término del error es $-\frac{h^2}{6} f'''(x_0)$ cuando $f'''(x_0) \neq 0$.

Dada una función $f : \mathbb{R} \rightarrow \mathbb{R}$ continuamente diferenciable, se desea aproximar su primera derivada. Por el teorema de Taylor aplicado a $x_0 + h$ y $x_0 - h$ alrededor de x_0 , obtenemos que:

1. **Expansión de $f(x_0 + h)$:**

$$f(x_0 + h) = f(x_0) + f'(x_0)h + \frac{f''(x_0)}{2!}h^2 + \frac{f'''(x_0)}{3!}h^3 + \dots$$

2. **Expansión de $f(x_0 - h)$:**

$$f(x_0 - h) = f(x_0) - f'(x_0)h + \frac{f''(x_0)}{2!}h^2 - \frac{f'''(x_0)}{3!}h^3 + \dots$$

Restando las expresiones anteriores, tenemos:

$$f(x_0 + h) - f(x_0 - h) = 2f'(x_0)h + \frac{2f'''(x_0)}{3!}h^3 + \dots$$

Dividimos la expresión resultante por $2h$:

$$\frac{f(x_0 + h) - f(x_0 - h)}{2h} = f'(x_0) + \frac{f'''(x_0)}{3!}h^2 + \dots$$

Finalmente, despejamos el término $f'(x_0)$:

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2h} - \frac{f'''(x_0)}{3!}h^2 - \dots$$

Por consiguiente, la aproximación de $f'(x_0)$ es:

$$f'(x_0) \approx \frac{f(x_0 + h) - f(x_0 - h)}{2h}.$$

Además, esta aproximación tiene un error de $O(h^2)$, y su primer término de error es:

$$-\frac{f'''(x_0)}{6}h^2.$$

✓ b) Adapte el programa de Matlab hecho en clase para visualizar el comportamiento del error de aproximación a medida que el paso h decrece desde $h = 10^{-1}, \dots, 10^{-16}$.

Adapte el programa de Matlab hecho en clase para visualizar el comportamiento del error de aproximación a medida que el paso h decrece desde $h = 10^{-1}$, ..., 10^{-16} .

A continuación se presenta un código que aproxima la derivada de $f(x) = \sin(x)$ en $x_0 = 1.2$ utilizando el método de diferencias finitas. Esta aproximación se realiza mediante la fórmula probada anteriormente. Además, el gráfico generado muestra cómo se comporta el error absoluto de esta aproximación al disminuir el paso h desde 10^{-1} hasta 10^{-16} .

```
!apt-get install -y octave
```

[Mostrar salida oculta](#)

```
%writefile primera_derivada.m
% aproximacion de la derivada usando el paso complejo
x0 = 1.2;
f0 = sin(x0);
fp = cos(x0);
i = -16:0.5:0;
h = 10.^i;
err01 = abs(fp - (sin(x0 + h) - sin(x0 - h)) ./ (2 * h)); % Error absoluto
d_err01 = fp/6*h.^2; % Error de truncamiento
loglog(h,d_err01,'r-.',h,err01,'-','LineWidth',2);
xlabel('h','fontsize',16);
ylabel('Error Absoluto','fontsize',16);
legend('error de truncamiento','diferencias finitas','location','southwest')
```

```
grid on;
```

```
% Guardar la gráfica como archivo PNG
```

```
print('derivada.png', '-dpng');
```

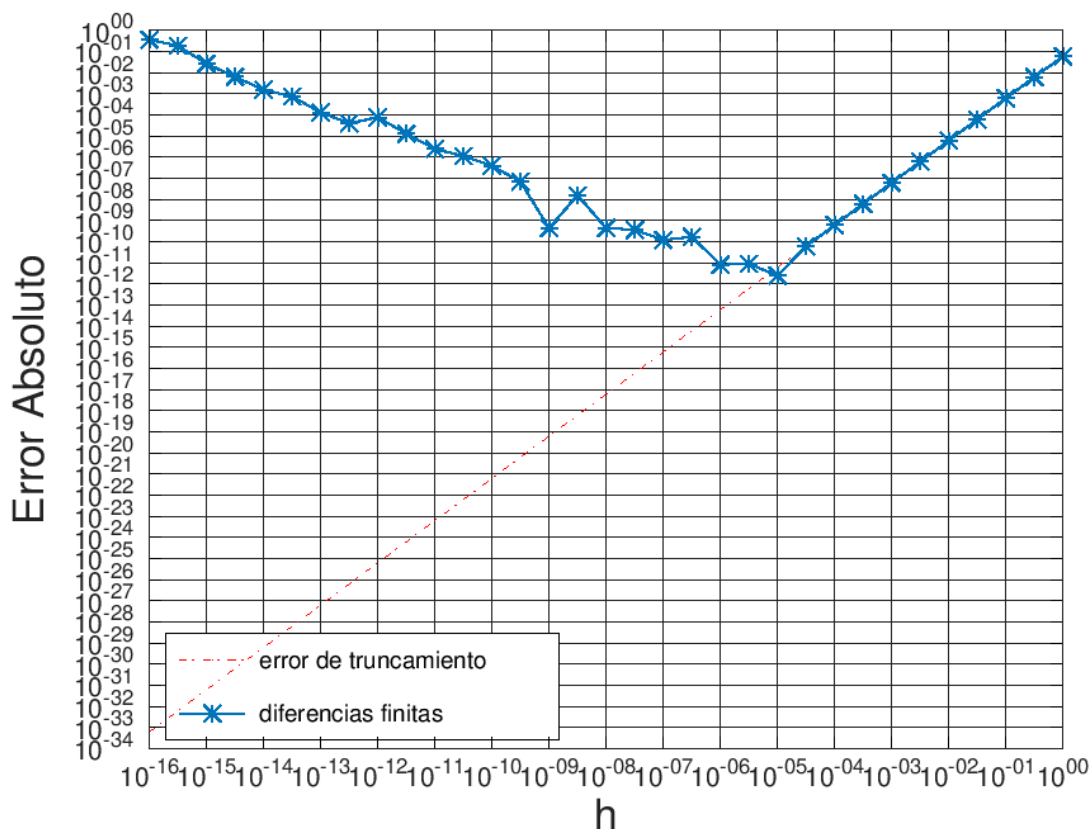
[Writing primera_derivada.m](#)

```
!octave primera_derivada.m
```

[Mostrar salida oculta](#)

```
from IPython.display import Image
Image(filename='derivada.png')
```

[Image](#)



✓ c) Muestre que

$$f'(x_0) = \frac{-f(x_0 + 2h) + 8f(x_0 + h) - 8f(x_0 - h) + f(x_0 - 2h)}{12h} + O(h^4)$$

De nuevo adapte su programa para visualizar el comportamiento del error de aproximación a medida que el paso h decrece desde $h = 10^{-1}, \dots, 10^{-16}$.

Dada una función $f: \mathbb{R} \rightarrow \mathbb{R}$ continuamente diferenciable, deseamos aproximar su primera derivada. Usamos el teorema de Taylor para expandir $f(x_0 + 2h)$, $f(x_0 + h)$, $f(x_0 - h)$ y $f(x_0 - 2h)$ alrededor de x_0 .

1. **Expansión de $f(x_0 + 2h)$:**

$$f(x_0 + 2h) = f(x_0) + f'(x_0)(2h) + \frac{f''(x_0)}{2!}(2h)^2 + \frac{f'''(x_0)}{3!}(2h)^3 + O(h^4)$$

2. **Expansión de $f(x_0 + h)$:**

$$f(x_0 + h) = f(x_0) + f'(x_0)(h) + \frac{f''(x_0)}{2!}(h)^2 + \frac{f'''(x_0)}{3!}(h)^3 + O(h^4)$$

3. **Expansión de $f(x_0 - h)$:**

$$f(x_0 - h) = f(x_0) - f'(x_0)(h) + \frac{f''(x_0)}{2!}(h)^2 - \frac{f'''(x_0)}{3!}(h)^3 + O(h^4)$$

4. **Expansión de $f(x_0 - 2h)$:**

$$f(x_0 - 2h) = f(x_0) - f'(x_0)(2h) + \frac{f''(x_0)}{2!}(2h)^2 - \frac{f'''(x_0)}{3!}(2h)^3 + O(h^4)$$

Restamos las expresiones adecuadamente para cancelar términos, y luego combinar los sumandos de orden superior.

1. **Restando $f(x_0 + 2h)$ y $f(x_0 - 2h)$:**

$$f(x_0 + 2h) - f(x_0 - 2h) = 4f'(x_0)h + \frac{16}{3!}f^{(3)}(x_0)h^3 + O(h^4).$$

2. **Restando $f(x_0 + h)$ y $f(x_0 - h)$:**

$$f(x_0 + h) - f(x_0 - h) = 2f'(x_0)h + \frac{4}{3!}f^{(3)}(x_0)h^3 + O(h^4).$$

Sumamos y restamos las expresiones anteriores con los coeficientes adecuados para obtener la aproximación deseada.

$$\frac{-f(x_0 + 2h) + 8f(x_0 + h) - 8f(x_0 - h) + f(x_0 - 2h)}{12h} = f'(x_0) + O(h^4).$$

Así, la aproximación de $f'(x_0)$ es:


$$f'(x_0) = \frac{-f(x_0 + 2h) + 8f(x_0 + h) - 8f(x_0 - h) + f(x_0 - 2h)}{12h} + O(h^4)$$

A continuación se presenta un código que aproxima la derivada de $f(x) = \sin(x)$ en $x_0 = 1.2$ utilizando el método de diferencias finitas. Esta aproximación se realiza mediante la fórmula probada anteriormente. Además, el gráfico generado muestra cómo se comporta el error absoluto de esta aproximación al disminuir el paso h desde 10^{-10} hasta 10^5 .

```
%writefile derivada_paso.m
% aproximacion de la derivada usando el paso complejo
x0 = 1.2;
f0 = sin(x0);
fp = cos(x0);
i = -10:0.5:5;
h = 10.^i;
err01 = abs(fp - (-sin(x0 + 2*h) + 8*sin(x0 + h) - 8*sin(x0 - h) - sin(x0 - 2*h)) ./ (12 * h));
d_err01 = fp/6*h.^2;
loglog(h,d_err01,'r-','h,err01','-','LineWidth',2);
xlabel('h','fontSize',16);
ylabel('Error Absoluto','fontSize',16);
legend('error de truncamiento','diferencias finitas','location','southwest')
```

grid on;

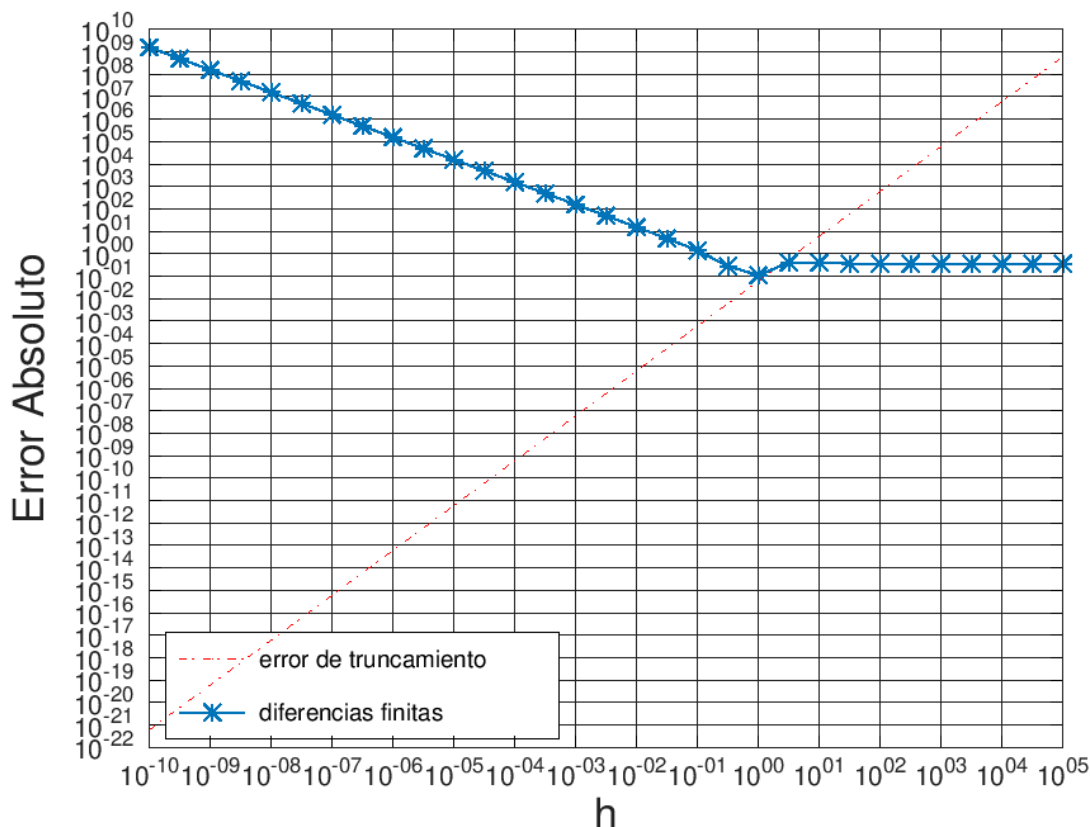
```
% Guardar la gráfica como archivo PNG
print('derivada.png', '-dpng');
```

 Writing derivada_paso.m

```
!octave derivada_paso.m
```

 [Mostrar salida oculta](#)

```
from IPython.display import Image
Image(filename='derivada.png')
```



- 6) Repaso O mayúscula y o minúscula de una función. Si $f(x) = O(x^2)$, $g(x) = O(x^3)$ y $h(x) = O(x^3)$ cuando $x \rightarrow 0$. ¿Cuáles de las siguientes afirmaciones son verdaderas? Justifique sus respuestas

Análisis de las afirmaciones:

Dado que:

- $f(x) = O(x^2)$ cuando $x \rightarrow 0$
- $g(x) = O(x^3)$ cuando $x \rightarrow 0$
- $h(x) = O(x^3)$ cuando $x \rightarrow 0$

Recordemos que:

- $f(x) = O(g(x))$ significa que existe una constante positiva M tal que $|f(x)| \leq M|g(x)|$ cerca de $x = 0$.
- $f(x) = o(g(x))$ significa que $\lim_{x \rightarrow 0} \frac{f(x)}{g(x)} = 0$.

Afirmación 1: $f(x) = o(x)$ cuando $x \rightarrow 0$

- Dado que $f(x) = O(x^2)$, existe una constante $M > 0$ tal que $|f(x)| \leq Mx^2$ cerca de $x = 0$.
- Evaluamos el límite:

$$\left| \frac{f(x)}{x} \right| = \frac{|f(x)|}{|x|} \leq \frac{Mx^2}{|x|} = M|x|$$

- Como $\lim_{x \rightarrow 0} M|x| = 0$, entonces:

$$\lim_{x \rightarrow 0} \left| \frac{f(x)}{x} \right| \leq \lim_{x \rightarrow 0} M|x| = 0$$

- Por lo tanto:

$$\lim_{x \rightarrow 0} \frac{f(x)}{x} = 0$$