Corrección de Examen

Tu Nombre

2024-12-10

ESCUELA POLITECNICA NACIONAL

Nombre:Stiv Quishpe

Link al respositorio

https://github.com/stiv001/CorrecionExamen1B.git

Pregunta 1

Suponga que dos puntos ((x_0, y_0)) y ((x_1, y_1)) se encuentran en línea recta con (y_1) no es igual a (y_0).
Existen dos fórmulas para encontrar la intersección (x) de la línea:

Método A:

$$x = \frac{y_0 x_1 - x_0 y_1}{y_1 - y_0}$$

Método B:

$$x = x_0 - \frac{(x_1 - x_0) y_0}{y_1 - y_0}$$

Usando los datos ((x_0, y_0) = (1.31, 3.24)) y ((x_1, y_1) = (1.93, 4.76)), determine el valor real de la intersección (x) (asumiendo redondeo a 6 cifras significativas):

(X)	1.31	1.93
(y)	3.24	4.76

Método A:

$$x = \frac{3.24 \cdot 1.93 - 1.31 \cdot 4.76}{4.76 - 3.24}$$
$$x = \frac{6.2532 - 6.2356}{1.52}$$
$$x = \frac{0.0176}{1.52} = 0.0115789$$

Método B:

$$x = 1.31 - \frac{(1.93 - 1.31) \cdot 3.24}{4.76 - 3.24}$$
$$x = 1.31 - \frac{0.62 \cdot 3.24}{1.52}$$
$$x = 1.31 - \frac{2.0088}{1.52}$$
$$x = 1.31 - 1.321578947 = 0.0115789$$

Los dos métodos dan el mismo valor para (x).

Usando aritmética de computadora con redondeo a 3 cifras significativas:

Método A:

$$x = \frac{3.24 \cdot 1.93 - 1.31 \cdot 4.76}{4.76 - 3.24}$$
$$x = \frac{6.25 - 6.22}{1.52} = 0.0197$$

Error relativo:

$$\text{error} = \frac{|0.0115789 - 0.0197|}{|0.0115789|} = 0.700$$

Método B:

$$x = 1.31 - \frac{(1.93 - 1.31) \cdot 3.24}{4.76 - 3.24}$$
$$x = 1.31 - 0.01 = 0.01$$

Error relativo:

$$\text{error} = \frac{|0.0115789 - 0.01|}{|0.0115789|} = 0.136$$

El **método B** es más preciso debido a que utiliza menos operaciones aritméticas.

Pregunta 2

Los primeros tres términos diferentes a cero de la serie de Maclaurin para la función arcotangente son:

$$x - \frac{1}{3}x^3 + \frac{1}{5}x^5$$

Calcule el error relativo en las siguientes aproximaciones de ( ) mediante el polinomio (en lugar del arcotangente).
Asuma que (= 3.14159).

Aproximación 1: 4 · arctan(1/2) + 4 · arctan(1/3)

$$\pi^* = 4\left(\left[\frac{1}{2} - \frac{1}{3}\left(\frac{1}{2}\right)^3 + \frac{1}{5}\left(\frac{1}{2}\right)^5\right] + \left[\frac{1}{3} - \frac{1}{3}\left(\frac{1}{3}\right)^3 + \frac{1}{5}\left(\frac{1}{3}\right)^5\right]\right)$$
$$\pi^* = 4\left(\frac{223}{480} + \frac{391}{1215}\right)$$
$$\pi^* = 4\left(\frac{6115}{7776}\right) = \frac{6115}{1944} = 3.1456$$

Error relativo:

$$\text{error} = \frac{|3.14159 - 3.1456|}{3.14159} = 0.0013$$

Aproximación 2: 16 · arctan(1/5) - 4 · arctan(1/239)

$$\pi^* = 16\left(\frac{1}{5} - \frac{1}{3}\left(\frac{1}{5}\right)^3 + \frac{1}{5}\left(\frac{1}{5}\right)^5\right) - 4\left(\frac{1}{239} - \frac{1}{3}\left(\frac{1}{239}\right)^3 + \frac{1}{5}\left(\frac{1}{239}\right)^5\right)$$

$$\pi^* = 16\left(\frac{9253}{46875}\right) - 4(0.004184076)$$

$$\pi^* = 3.158357333 - 0.016736304 = 3.1416$$

Error relativo:

$$\text{error} = \frac{|3.14159 - 3.1416|}{3.14159} = 0.0000098$$

Pregunta 3

```python
def secant_method(f, x0, x1, tol=1e-6, max_iter=100):

    x_prev = x0
    x_curr = x1
    iter_count = 0
    f_x_prev = f(x_prev)
    f_x_curr = f(x_curr)

    while abs(f_x_curr) > tol and iter_count < max_iter:
        # Verifica que no haya división por cero
        if f_x_curr == f_x_prev:
            raise ValueError("División por cero: f(x_curr) y f(x_prev) son iguales.")

        # Calcula el siguiente punto
        x_next = x_curr - f_x_curr * (x_curr - x_prev) / (f_x_curr - f_x_prev)

        # Actualiza las variables para la próxima iteración
        x_prev, f_x_prev = x_curr, f_x_curr
        x_curr, f_x_curr = x_next, f(x_next)

        iter_count += 1

    if abs(f_x_curr) <= tol:
        return x_curr, iter_count
    else:
```

```
        raise ValueError("El método no convergió en el número máximo de iteraciones.")
```

Ejemplo1

```
i = 0

def func(x):
    global i
    i += 1
    y = x**3 - 3 * x**2 + x - 1
    print(f"Llamada i={i}\t x={x:.5f}\t y={y:.2f}")
    return y


secant_method(func, x0=2, x1=3)
```

```
Llamada i=1  x=2.00000   y=-3.00
Llamada i=2  x=3.00000   y=2.00
Llamada i=3  x=2.60000   y=-1.10
Llamada i=4  x=2.74227   y=-0.20
Llamada i=5  x=2.77296   y=0.03
Llamada i=6  x=2.76922   y=-0.00
Llamada i=7  x=2.76929   y=-0.00
Llamada i=8  x=2.76929   y=0.00

(2.7692923542484045, 6)
```

Ejemplo2

```
i = 0
import math

def func(x):
    global i
    i += 1
```

```
    y = math.sin(x) + 0.5
    print(f"Llamada i={i}\t x={x:.5f}\t y={y:.2f}")
    return y


secant_method(func, x0=2, x1=3)
```

```
Llamada i=1  x=2.00000   y=1.41
Llamada i=2  x=3.00000   y=0.64
Llamada i=3  x=3.83460   y=-0.14
Llamada i=4  x=3.68602   y=-0.02
Llamada i=5  x=3.66399   y=0.00
Llamada i=6  x=3.66520   y=-0.00
Llamada i=7  x=3.66519   y=-0.00

(3.66519143172732, 5)
```

Pregunta 4

```
import math

def bisection_method(f, a, b, tol=1e-6, max_iter=100):
    if f(a) * f(b) > 0:
        raise ValueError("El intervalo no tiene cambio de signo.")

    iter_count = 0
    while (b - a) / 2 > tol and iter_count < max_iter:
        c = (a + b) / 2
        if f(c) == 0:  # Raíz exacta encontrada
            return c
        elif f(a) * f(c) < 0:
            b = c
        else:
            a = c
        iter_count += 1

    return (a + b) / 2

def f(x):
```

```
    return math.sin(x)

intervals = [
    (-1, 2),
    (-5, 4),
    (-2.5, -1),
    (-4, 5),
    (3, 5),
    (-3.5, 3),
]

for a, b in intervals:
    try:
        root = bisection_method(f, a, b)
        print(f"Intervalo [{a}, {b}]: Raíz encontrada  {root}")
    except ValueError as e:
        print(f"Intervalo [{a}, {b}]: {e}")
```

```
Intervalo [-1, 2]: Raíz encontrada  -2.384185791015625e-07
Intervalo [-5, 4]: Raíz encontrada  -3.14159232378006
Intervalo [-2.5, -1]: El intervalo no tiene cambio de signo.
Intervalo [-4, 5]: Raíz encontrada  3.14159232378006
Intervalo [3, 5]: Raíz encontrada  3.1415929794311523
Intervalo [-3.5, 3]: El intervalo no tiene cambio de signo.
```

Pregunta 5

```
def newton_method(f, df, x0, tol=1e-6, max_iter=100):
    iter_count = 0
    x_curr = x0

    while iter_count < max_iter:
        f_x = f(x_curr)
        df_x = df(x_curr)

        if abs(df_x) < 1e-12:
            return "Error (división por 0)"

        x_next = x_curr - f_x / df_x
```