Ayudantía

SCT - Semana - 03

IPF y Método de Bisección

1. Se tiene la siguiente función continua y diferenciable en $\mathbb{R}$:

$$f(x) = x^3 + 4x^2 - 10$$

a) Suponga que existe una raíz de $f(x)$ en $[1, 2]$. Indique cuántas iteraciones requiere el método de la bisección para aproximar la raíz con 10 decimales correctos.

- Para aproximar la raíz con 10 decimales correctos, el error del resultado final debe ser menor a $\dfrac{10^{-10}}{2}$.
- Si, en la n-ésima iteración, el método de la bisección termina en el intervalo $[a_n, b_n]$ y retorna el valor $\dfrac{a_n + b_n}{2}$ como aproximación, entonces el error es $\dfrac{b_n - a_n}{2}$ para esta iteración, donde $b_n - a_n$ es el tamaño del intervalo $[a_n, b_n]$, Dado que, en cada iteración, el tamaño del intervalo se reduce a la mitad, después de n iteraciones, el tamaño resultante es $b_n - a_n = \dfrac{b - a}{2^n}$, donde $b - a$ es el tamaño del intervalo inicial $[a, b]$ para la bisección. Es decir, el error es $\dfrac{b_n - a_n}{2} = \dfrac{\frac{b-a}{2^n}}{2} = \dfrac{b - a}{2^{n+1}}$. Como $[a, b] = [1, 2]$, el error es $\dfrac{1}{2^{n+1}}$.

Juntando ambas definiciones, la cantidad n de iteraciones para aproximar la raíz con 10 decimales correctos se obtiene de la siguiente inecuación:

$$\frac{1}{2^{n+1}} < \frac{10^{-10}}{2}$$

Multiplicando ambos lados por 2^{n+1} y por 10^{10}:

$$10^{10} < 2^n$$

Dado que $2^{10} = 1024 \approx 10^3$, se puede decir que $2^{30} \approx 10^9$, de donde se puede estimar que el primer entero n que cumple la inecuación es $n = 34$, pues $2^{34} = 2^4 \times 2^{30} = 16 \times 2^{30} \approx 16 \times 10^9 > 10^{10}$.
Siendo más precisos, de la inecuación

$$2^n > 10^{10}$$

se obtiene $n > \log_2(10^{10}) = 10 \log_2(10) \approx 10 \times 3.322 = 33.22$. El primer entero mayor a 33.22 es $n = 34$.

b) Aplique computacionalmente el método de la bisección en $[1, 2]$ para verificar su respuesta anterior.

```python
def n_pasos_biseccion(f, a, b, tol):
    n = 0

    # tol seria 10^(-5) / 2
    # Tambien puedes usar
    # while abs(b - a) > tol:
    # sin divisiones por 2 y usar simplemente tol = 10^(-5)
    while abs(b - a) / 2 > tol:
        n += 1
        c = (a + b) / 2
        if f(c) == 0:
            return c, n

        if f(c) * f(a) < 0:
            b = c
        else:
            a = c

    return (a + b) / 2, n
```

```python
>>> f = lambda x: x*x*x + 4*x*x - 10
>>> a = 1
>>> b = 2
```

```
>>> tol = 1e-5 / 2  # 1e-5 representa 1 * 10^(-5)
>>>
>>> resultado, n = n_pasos_biseccion(f, a, b, tol)
>>> n
34
```

2. Sean $f(t)$ y $g(t)$ dos funciones reales continuas definidas para todo $t > 0$. Construya un algoritmo que determine α tal que $g(\alpha) + f\left(\frac{1}{\alpha}\right) = A$, donde $0 < A < 1$. Considere que $g(1) + f(1) = 0$ y $g(10^4) + f(10^{-4}) = 1$. El algoritmo debe recibir A y n como parámetros y retornar α con n decimales de precisión.

Explique claramente todos los supuestos que considera.

La búsqueda de α tal que $g(\alpha) + f\left(\frac{1}{\alpha}\right) = A$ es un **problema de búsqueda de raíces**. En este problema, se debe **definir la función** h a la cual queremos buscarle una raíz, es decir, un valor α tal que $h(\alpha) = 0$.

La ecuación $g(\alpha) + f\left(\frac{1}{\alpha}\right) = A$ puede reescribirse, restando A a ambos lados, como

$$g(\alpha) + f\left(\frac{1}{\alpha}\right) - A = 0$$

y, desde ahí, se puede definir una función h tal que

$$h(t) = g(t) + f\left(\frac{1}{t}\right) - A.$$

Si α es la solución de la ecuación original, entonces se puede verificar que $h(\alpha) = g(\alpha) + f\left(\frac{1}{\alpha}\right) - A = 0$, es decir, α es una raíz de h.

La información adicional del contexto nos permite decidir qué método de búsqueda de raíces deberíamos aplicar. En este caso, hay cuatro piezas importantes de información contextual:

a) $g(10^4) + f(10^{-4}) = 1$.
El valor 10^{-4} equivale a $\frac{1}{10^4}$, volviendo lo anterior en $g(10^4) + f\left(\frac{1}{10^4}\right) = 1$, algo muy similar a la forma $g(t) + f\left(\frac{1}{t}\right)$ de la ecuación original y la función h definida. Esto nos da la idea de evaluar $h(10^4)$:

$$\begin{aligned} h(10^4) &= g(10^4) + f\left(\frac{1}{10^4}\right) - A \\ &= 1 - A \end{aligned}$$

b) $g(1) + f(1) = 0$.
De nuevo, como $1 = \frac{1}{1}$, se puede hacer algo similar a lo anterior y evaluar $h(1)$:

$$\begin{aligned} h(1) &= g(1) + f\left(\frac{1}{1}\right) - A \\ &= 0 - A \\ &= -A \end{aligned}$$

c) $0 < A < 1$.
Debido a que A es positivo, $h(1) = -A$ es negativo.
Debido a que A es menor a 1, $h(10^4) = 1 - A$ es positivo.

d) f y g son funciones continuas.
Por álgebra de continuidad, esto vuelve a h una función continua.
El hecho de que h sea continua implica que, si $h(1)$ es negativo y $h(10^4)$ es positivo, entonces, en algún punto t entre 1 y 10^4, $h(t)$ debe valer 0 debido al teorema de Bolzano. En otras palabras, sabemos que existe algún valor $\alpha \in [1, 10^4]$ tal que $h(\alpha) = 0$.

Todo lo anterior nos indica que podemos usar el **método de la bisección** sobre la función $h(t) = g(t) + f\left(\frac{1}{t}\right) - A$, partiendo repetidamente por la mitad el intervalo $[a, b] = [1, 10^4]$ hasta tener n decimales de precisión, es decir, hasta la iteración k donde $\frac{b_k - a_k}{2} < \frac{10^{-n}}{2}$ o, equivalentemente, $b_k - a_k < 10^{-n}$.