Facultad de Matemática, Astronomía, Física y Computación, U.N.C.

Computación

Aula Virtual: https://famaf.aulavirtual.unc.edu.ar/course/view.php?id=747

Resguardo tutoriales: https://www.famaf.unc.edu.ar/~moreschi/docencia/Computacion/

Tutorial Problema 5 de la Guía N° 3

Problema 5:

Sea $f : [a, b] \to \mathbb{R}$ una función contínua tal que $f(a)f(b) < 0$. Escriba un programa en PYTHON, en forma de script, que implemente el método de bisección para hallar una raíz $r \in (a, b)$ de $f(x)$. El programa debe detenerse cuando la aproximación x_n de r obtenida al cabo de n iteraciones satisfaga las condiciones: $|x_n - x_{n-1}| < \delta$ y $|f(x_n)| < \varepsilon$, donde $\varepsilon > 0$ y $\delta > 0$ son tolerancias dadas. En este caso el programa debe imprimir en pantalla la aproximación a la raíz y el número de iteraciones realizadas. Si al cabo de M iteraciones la aproximación obtenida no satisface la tolerancia deseada, el programa debe detenerse indicando que no se obtuvo una aproximación satisfactoria. Antes de iniciar la iteración el programa debe verificar la condición inicial $f(a)f(b) < 0$, si esta no se cumple, el programa debe terminar con un mensaje de error que indique esta situación.

Utilizando $M = 100$, $\varepsilon = 10^{-6}$, y $\delta = 10^{-6}$, calcule raíces positivas de las siguientes funciones e intervalos

a) $f(x) = 3x - \tan(x)$, en $[0.1, 1.5]$.

b) $f(x) = x^6 - 5x^5 + x^4 - 3x^3 + x^2 - x + 1$, en $[0, 1]$.

Grafique estas funciones en sus respectivos intervalos; mostrando los mismos en pantalla y guardándolos en archivos '.png' en subdirectorio apropiado (ej.: "gráficos/").

Tutorial:

- Guarde en el archivo p5-g3.py las siguientes instrucciones:

```
"""
 En esta versión usa la función para el método
 de la bisección.
"""
import numpy as np
import matplotlib.pyplot as plt

#-------------------------------------
# Función parte a), cuya raíz queremos hallar
def f1(x):
    return 3*x - np.tan(x)

# Función parte b), cuya raíz queremos hallar
def f2(x):
    return x**6 - 5*x**5 + x**4 - 3*x**3 + x**2 - x + 1


def biseccion(f,a,b,tolx,toly,maxn):
```

```
    print()
    if (f(a)*f(b) > 0):
        print("************************************************")
        print("* f(a) y f(b) deben tener distinto signo.      ")
        print("************************************************")
        return

    nmax = int((np.log(b-a)-np.log(tolx))/np.log(2.)) + 1
    if (maxn > nmax):
        print("************************************************")
        print("* Dado el intervalo, la precisión en x se puede ")
        print("* alcanzar con nmax =",nmax)
        print("************************************************")
    intervalo = b - a
    c = a + 0.5*intervalo
    n = 0
    while ((intervalo > tolx) or ( abs(f(c)) > toly) and (n<=maxn)) :
        n += 1
        c = a + 0.5*intervalo

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

        intervalo = b - a

    # Aproximación a la raíz (punto medio del intervalo final)
    c1 = a + 0.5*intervalo

    print("\nLa solución es: %27.20e" % (c1))
    print("\nEl programa realizó %d iteraciones de bisección\n" % (n))
    print("      intervalo = %27.20e" % (intervalo) )
    print("      abs(f(c)) = %27.20e" % (abs(f(c1)) ) )
    print()
    return c1


# Caso a) ----------------------------------------
# intervalo
a = 0.1
b = 1.5

t1 = np.arange(a, b, (b-a)/150)  # para el gráfico

# Tolerancia para el intervalo que contiene la raiz
epsilon = 1.0e-6
delta   = 1.0e-6

# máximo número de pasos
M = 100

x1=biseccion(f1,a,b,epsilon,delta,M)

print(' x =',x1)

```

```
# Caso b) ----------------------------------------
# intervalo

a = 0.
b = 1.

t2 = np.arange(a, b, (b-a)/150)  # para el gráfico

# Tolerancia para el intervalo que contiene la raiz
epsilon = 1.0e-6
delta   = 1.0e-6
M = 100

x2=biseccion(f2,a,b,epsilon,delta,M)

print(' x =',x2)

#-------------------------------------------------
"""
   Hagamos un gráfico para 'ver' las raíces.
"""

plt.figure( figsize=(12, 8))
plt.subplot(211)
plt.title('3*x - tan(x)')
plt.plot(t1, f1(t1))
plt.grid()
plt.axhline(y=0, color='#000000') # línea horizontal para ver la raíz
plt.axvline(x=x1, ymin=0.05, ymax=0.95, linewidth=2, color='#d62728')

plt.subplot(212)
plt.title('x**6 - 5*x**5 + x**4 - 3*x**3 + x**2 - x + 1')
plt.plot(t2, f2(t2))
plt.grid()
plt.axhline(y=0, color='#000000') # línea horizontal para ver la raíz
plt.axvline(x=x2, ymin=0.05, ymax=0.95, linewidth=2, color='#d62728')

plt.subplots_adjust(hspace=0.5)
plt.savefig('graficos/p5-g3.png', dpi=120)  # guarda figura en un archivo

plt.show()
```

- Desde la terminal ejecute:

  python3 p5-g3.py

  e interprete el resultado.

Modifique el programa y pruebe otras cosas.