LAB8 - Rozwiązywanie równań nieliniowych

Gosztyła Mikołaj, Smółka Antoni

Zadanie 1

Dla poniższych funkcji i punktów początkowych metoda Newtona zawodzi. Wyjaśnij dlaczego. Następnie znajdź pierwiastki, modyfikując wywołanie funkcji `scipy.optimize.newton` lub używając innej metody.

(a) $ f(x) = x^3 - 5x , x_0 = 1 $

(b) $ f(x) = x^3 - 3x + 1 , x_0 = 1 $

(c) $ f(x) = 2 - x^5 , x_0 = 0.01 $

(d) $ f(x) = x^4 - 4.29x^2 - 5.29 $, $ x_0 = 0.8 $

(a) $ f(x) = x^3 - 5x, x_0 = 1 $ Wyjaśnienie zawodzenia: Metoda Newtona zawodzi, gdy pochodna funkcji w punkcie początkowym jest równa zero lub bardzo bliska zera, co może prowadzić do dużych skoków lub braku konwergencji. W tym przypadku, pochodna funkcji $f'(x)=3x^2-5$. Dla $x_0=1$, $f'(1)=3(1)^2-5=-2$, co jest wartością znacznie różną od zera, więc przyczyna może leżeć gdzie indziej, na przykład w złożonym krajobrazie funkcji wokół punktu startowego.

(b) $ f(x) = x^3 - 3x + 1, x_0 = 1 $ Wyjaśnienie zawodzenia: Podobnie jak wyżej, sprawdzamy pochodną: $f'(x)=3x^2-3$. Dla $x_0=1$, $f'(1)=3(1)^2-3=0$. W tym przypadku, metoda zawodzi, ponieważ pochodna jest równa zero, co uniemożliwia wykonanie kolejnego kroku w metodzie Newtona (dzielenie przez zero).

(c) $ f(x) = 2 - x^5, x_0 = 0.01 $ Wyjaśnienie zawodzenia: Pochodna tej funkcji to $f'(x)=-5x^4$. Dla bardzo małego $x_0=0.01$, wartość $f'(0.01)$ jest bardzo bliska zeru ($f'(0.01)=-5(0.01)^4=-0.0000005$), co sprawia, że metoda jest niestabilna i niekonwergentna.

(d) $ f(x) = x^4 - 4.29x^2 - 5.29, x_0 = 0.8 $ Wyjaśnienie zawodzenia: Pochodna funkcji wynosi $f'(x)=4x^3-8.58x$. Dla $x_0=0.8$, $f'(0.8)=4(0.8)^3-8.58(0.8)=1.6384$. Zawodzenie może być spowodowane brakiem konwergencji w obszarze lokalnych minimów lub maksimów funkcji, które nie są pierwiastkami.

Podpunkt bc i jest git, ale a i d to nie jestem pewien tych odpowiedzi.

```python
import scipy.optimize as opt

# Definicje funkcji
def func_a(x):
    return x**3 - 5*x

def func_b(x):
    return x**3 - 3*x + 1
```

```
def func_c(x):
    return 2 - x**5

def func_d(x):
    return x**4 - 4.29*x**2 - 5.29

def find_root(func, x0, method='newton'):
    try:
        # Próbujemy standardową metodę Newtona
        root = opt.newton(func, x0)
    except RuntimeError:
        # Gdy metoda Newtona zawodzi, stosujemy metodę brentq, która
wymaga podania zakresu
        # Zakres dobieramy eksperymentalnie na podstawie wizualnej
analizy funkcji
        root = opt.brentq(func, x0 - 1, x0 + 1)
    return root

root_a = find_root(func_a, 1)
root_b = find_root(func_b, 1)
root_c = find_root(func_c, 0.01)
root_d = find_root(func_d, 0.8)

print("Pierwiastek funkcji a):", root_a)
print("Pierwiastek funkcji b):", root_b)
print("Pierwiastek funkcji c):", root_c)
print("Pierwiastek funkcji d):", root_d)
```

```
Pierwiastek funkcji a): 4.744493466790075e-24
Pierwiastek funkcji b): 1.0000007188230098
Pierwiastek funkcji c): 0.01
Pierwiastek funkcji d): -0.7870232540616441
```

Zadanie 2

Dane jest równanie:

$$f(x)=x^2-3x+2=0$$

Każda z następujących funkcji definiuje równoważny schemat iteracyjny:

$$g_1(x)=\frac{x^2+2}{3}$$

$$g_2(x)=\sqrt{3x-2}$$

$$g_3(x)=3-\frac{2}{x}$$

$$g_4(x) = \frac{x^2 - 2}{2x - 3}$$

(a) Przeanalizuj zbieżność oraz rząd zbieżności schematów iteracyjnych odpowiadających funkcjom $ g_i(x) $ dla pierwiastka $ x = 2 $ badając wartość $ |g'_i(2)| $.

(b) Potwierdź analizę teoretyczną implementując powyższe schematy iteracyjne i weryfikując ich zbieżność (lub brak). Każdy schemat iteracyjny wykonaj przez 10 iteracji.

Wyznacz eksprymentalnie rząd zbieżności każdej metody iteracyjnej ze wzoru:

$ r = \frac{\ln \left(\frac{\epsilon_k}{\epsilon_{k+1}}\right)}{\ln \left(\frac{\epsilon_{k-1}}{\epsilon_k}\right)} $

gdzie błąd bezwzględny $\epsilon_k $ definiujemy jako $ \epsilon_k = |x_k - x^/$*, $ x_k $ jest przybliżeniem pierwiastka w k-tej iteracji, a x^* dokładnym położeniem pierwiastka równania.

(c) Na wspólnym rysunku przedstaw wykresy błędu względnego każdej metody w zależności od numeru iteracji. Użyj skali logarytmicznej na osi y (pomocna będzie funkcja `semilogy`).

Stwórz drugi rysunek, przedstawiający wykresy błędu względnego tylko dla metod zbieżnych.

Zadanie 3

Napisz schemat iteracji wg metody Newtona dla każdego z następujących równań nieliniowych:

(a) $ x^3 - 2x - 5 = 0 $

(b) $ e^{-x} = x $

(c) $ x \sin(x) = 1 $.

Jeśli $ x_0 $ jest przybliżeniem pierwiastka z dokładnością 4 bitów, ile iteracji należy wykonać aby osiągnąć:

- 24-bitową dokładność
- 53-bitową dokładność?

```python
import math

def newton_method(f, df, x0, epsilon, max_iter):
    x = x0
    for n in range(max_iter):
        fx = f(x)
        dfx = df(x)
        if abs(dfx) < epsilon:  # Aby uniknąć dzielenia przez zero
            print("Derivative too small.")
            break
        x_new = x - fx / dfx
        if abs(x_new - x) < epsilon:  # Sprawdzenie zbieżności
            return x_new, n
        x = x_new
    return x, max_iter
```

```python
def f_a(x):
    return x**3 - 2*x - 5

def df_a(x):
    return 3*x**2 - 2

def f_b(x):
    return math.exp(-x) - x

def df_b(x):
    return -math.exp(-x) - 1

def f_c(x):
    return x * math.sin(x) - 1

def df_c(x):
    return math.sin(x) + x * math.cos(x)

x0 = 1
epsilon = 2**-54
max_iter = 100

root_a, iterations_a = newton_method(f_a, df_a, x0, epsilon, max_iter)
print("Root a:", root_a, "Iterations:", iterations_a)

root_b, iterations_b = newton_method(f_b, df_b, x0, epsilon, max_iter)
print("Root b:", root_b, "Iterations:", iterations_b)

root_c, iterations_c = newton_method(f_c, df_c, x0, epsilon, max_iter)
print("Root c:", root_c, "Iterations:", iterations_c)
```

```
Root a: 2.0945514815423265 Iterations: 9
Root b: 0.5671432904097838 Iterations: 4
Root c: 1.1141571408719302 Iterations: 100
```

Zadanie 4

Napisz schemat iteracji wg metody Newtona dla następującego układu równań nieliniowych:

$$x_1^2 + x_2^2 = 1$$

$$x_1^2 - x_2 = 0$$

Korzystając z faktu, że dokładne rozwiązanie powyższego układu równań to:

$ x_1 = \pm \sqrt{\frac{\sqrt{5}}{2} - \frac{1}{2}} $

$ x_2 = \frac{\sqrt{5}}{2} - \frac{1}{2} $

oblicz błąd względny rozwiązania znalezionego metodą Newtona.