Obliczenia naukowe
Lista 3

Szymon Janiak

November 19, 2023

Zadanie 1

Opis problemu

Napisać funkcję rozwiązującą równanie $f(x) = 0$ metodą bisekcji.

Dane wejściowe

- `f` — funkcja f w postaci anonimowej funkcji
- `a,b` — liczby typu `Float64` określające końce przedziału początkowego
- `delta, epsilon` — liczby typu `Float64` określające dokładności obliczeń

Czwórka wartości `(r, val, it, err)`.

- `r` — przybliżenie pierwiastka równania $f(x) = 0$
- `v` — wartość funkcji w r
- `it` — liczba wykonanych iteracji
- `err` — sygnalizacja błędu, możliwe wartości:
  - `0` — brak błędu
  - `1` — funkcja nie zmienia znaku w przedziale $[a; b]$

Opis algorytmu

Metoda bisekcji polega na stopniowym zawężaniu przedziału szukania naszego pierwiastka do momentu, gdy nasz wynik będzie wystarczająco bliski, co do zależne jest od zdefiniowanego przez nas ϵ. Do zastosowania tej metody potrzebne są spełnione dwa założenia:

- funkcja $f(x)$ jest ciągła w przedziale domkniętym $[a, b]$,
- funkcja przyjmuje różne znaki na końcach przedziału.

Kolejne iteracje algorytmu przesuwają jedną z granic przedziału bliżej pierwiastka o połowę długości przedziału. Wybór końca do przesunięcia polega na sprawdzeniu które przesunięcie zwróci nam przedział, który spełnia warunek zmiany znaku w naszym nowym przedziale.

Rozwiązanie

Algorithm 1 bisection method

$val \leftarrow 0$
$it \leftarrow 0$
$e \leftarrow b - a$
$u \leftarrow f(a)$
$v \leftarrow f(b)$
$r \leftarrow \frac{1}{2} * (a + b)$
if $sign(u) = sign(v)$ **then**
    $err \leftarrow 1$
    return r, val, it, err
end if
while $abs(e) > \epsilon$ and $abs(f(r)) > \delta$ **do**
    $e \leftarrow frace2$
    $r \leftarrow a + e$
    $val \leftarrow f(r)$
    $it \leftarrow it + 1$
    if $abs(e) < \delta$ or $abs(val) < \epsilon$ **then**
        return r, val, it, err
    end if
    if $sign(val) \neq sign(u)$ **then**
        $g \leftarrow r$
        $v \leftarrow val$
    else
        $a \leftarrow r$
        $u \leftarrow val$
    end if
end while
return r, val, it, err

Zadanie 2

Opis problemu

Napisać funkcję rozwiązującą równanie $f(x) = 0$ metodą Newtona.

Dane wejściowe

- `f` — funkcja f w postaci anonimowej funkcji
- `pf` — pochodna funkcji f w postaci anonimowej funkcji
- `x0` — przybliżenie początkowe
- `delta, epsilon` — liczby typu `Float64` określające dokładności obliczeń
- `maxit` — liczba całkowita określająca dopuszczalną liczbę iteracji

Dane wyjściowe

Czwórka wartości `(r, v, it, err)`.

- `r` — przybliżenie pierwiastka równania $f(x) = 0$
- `v` — wartość funkcji w r
- `it` — liczba wykonanych iteracji
- `err` — sygnalizacja błędu, możliwe wartości:
  - `0` — metoda zbieżna
  - `1` — nie osiągnięto wymaganej dokładności w `maxit` iteracji
  - `2` — pochodna bliska zeru

Opis algorytmu

W tym algorytmie potrzebujemy założyć, że w naszym przedziale $[a, b]$ znajduje sie dokładnie jeden pierwiastek funkcji f, oraz różne znaki funkcji na krańcach przedziału. Dodatkowym wymaganiem jest stały znak pierwszej i drugiej pochodnej funkcji w tym przedziale. Na początku przyjmujemy sobie za x_1 granicę a lub b i wyznaczamy równanie stycznej do wykresu funkcji w punkcie $[x_1, f(x_1)]$, następnie wyznaczamy odciętą x_2 punktu przecięcia tej stycznej z osią OX - w ten sposób otrzymujemy kolejne przybliżenie naszego rozwiązania.Całą procedurę powtarzamy do momentu gdy otrzymamy wynik mieszczący się w naszym ϵ tworząc kolejne styczne.

Rozwiązanie

Algorithm 2 Newton method

$val \leftarrow f(x_0)$
$val_prime \leftarrow 0$
$x_1 \leftarrow 0$
$it \leftarrow 1$
if $abs(v) < \epsilon$ **then**
    $err \leftarrow 0$
    return x_0, val, it, err
end if
for it to $maxit$ **do**
    $val_prime \leftarrow pf(x_0)$
    $x_1 = x_0 - fracvalval_prime$
    $val \leftarrow f(x_1)$
    if $abs(val_prime) \leq NEARZERO$ or $isInf(abs(val_prime))$ **then**
        $err \leftarrow 2$
        return $x_0, f(x_0), it, err$
    end if
    if $abs(x_1 - x_0) < \delta$ or $abs(val) < \epsilon$ **then**
        return return x_1, val, it, err
    end if
    $x_0 \leftarrow x_1$
end for
$err \leftarrow 1$
return x_0, val, it, err

Zadanie 3

Opis problemu

Napisać funkcję rozwiązującą równanie $f(x) = 0$ metodą siecznych.

Dane wejściowe

- `f` — funkcja f w postaci anonimowej funkcji
- `x0,x1` — przybliżenia początkowe
- `delta, epsilon` — liczby typu `Float64` określające dokładności obliczeń
- `maxit` — liczba całkowita określająca dopuszczalną liczbę iteracji

Dane wyjściowe

Czwórka wartości `(r,v,it,err)`.

- `r` — przybliżenie pierwiastka równania $f(x) = 0$
- `v` — wartość funkcji w r
- `it` — liczba wykonanych iteracji
- `err` — sygnalizacja błędu, możliwe wartości:
  - `0` — metoda zbieżna
  - `1` — nie osiągnięto wymaganej dokładności w `maxit` iteracji

Opis algorytmu

Do użycia tej metody potrzebujemy dwa punkty startowe x_0 i x_1, których będziemy używać do wyznaczania następnych przybliżeń. Obliczamy $f(x_0)$ i $f(x_1)$, a następnie poprowadzamy przez te dwa punkty sieczną. Przecięcie siecznej z osią OX wyznaczy nam punkt do następnej iteracji. W ten sposób będziemy wyznaczać coraz to bliższe przybliżenia funkcji używając do tego zawsze dwóch wartości x_n oraz x_{n+1}. Metoda ta nie zawsze jest zbieżna.

Rozwiązanie

Algorithm 3 secant method

$it \leftarrow 0$
$a \leftarrow x_0$
$b \leftarrow x_1$
$val \leftarrow f(x_0)$
$valnext \leftarrow f(x_1)$
for it to $maxit$ **do**
  if $abs(val) > abs(valnext)$ **then**
    $a, b = b, a$
    $val, valnext = valnext, val$
  end if
  $d \leftarrow \frac{b-a}{valnext-val}$
  $b \leftarrow a$
  $valnext \leftarrow val$
  $a \leftarrow a - d * val$
  $val \leftarrow f(a)$
  if $abs(val) < \epsilon$ or $abs(b-a) < delta$ **then**
    return a, val, it, err
  end if
end for
$err \leftarrow 1$
return a, val, it, err