Projekt nr 15: Metoda Newtona i zmodyfikowana metoda Newtona wyznaczania zer pojedynczych i podwójnych funkcji f: R $\rightarrow$ R

Michał Kukla
Grupa 4

10 maja 2023

1 Opis matematyczny

1.1 Idea

Załóżmy, że f jest funkcją zmiennej rzeczywistej, oraz, że f : R $\rightarrow$ R. Zadaniem, którym się zajmiemy, będzie znalezienie takiej wartości $\alpha \in R$, że f(α) = 0.

W praktyce rzadko mamy do czynienia z funkcjami, których miejsca zerowe da się znaleźć w sposób analityczny, używając skończonej liczby operacji arytmetycznych. Na przykład, wyznaczamy pierwiastki wielomianów drugiego stopnia w sposób łatwy, czego nie da się zrobić w przypadku wielomianów wyższych stopni. Co więcej, dla wielomianów stopnia 5 i większego, nie istnieją ogólne analityczne wzory określające ich pierwiastki na podstawie współczynników. Oczywiście, pierwiastki pewnych konkretnych wielomianów stopnia wyższego niż 4 da się znaleźć analitycznie (np. dla f(x) = $x^5 - x$), ale w ogólnym przypadku jest to niemożliwe w skończonej liczbie operacji arytmetycznych. Można wyznaczyć pierwiastki w sposób przybliżony, czyli z pewną dokładnością, stosując metody iteracyjne. Polegają one na tym, że startując z danego przybliżenia początkowego x0 $\in$ R tworzymy ciąg kolejnych przybliżeń x_k, taki, że $x_k \rightarrow \alpha$ przy k $\rightarrow$ inf.

1.2 Metoda Newtona

Niech f : R $\rightarrow$ R będzie funkcją klasy C^2 oraz niech x0 $\in$ R będzie danym przybliżeniem początkowym. Przy zapisaniu funkcji f w szereg Taylora w otoczeniu x0 i uwzględnieniu tylko dwóch wyrazów, otrzymamy przybliżenie funkcji f wielomianem stopnia 1. Oznaczmy go przez p. Mamy:

$$p(x) = f(x_k) + f'(x_k) * (x - x_k) \tag{1}$$

Jest to styczna do wykresu funkcji f poprowadzona w punkcie $(x_k, f(x_k))$ W metodzie Newtona, kolejne przybliżenie, czyli x_{k+1}, jest miejscem zerowym stycznej p. Wyznaczmy je. Skoro x_{k+1} jest miejscem zerowym p, to $p(x_{k+1}) = 0$. Mamy zatem:

$$0 = p(x_{k+1}) = f(x_k) + f'(x_k)(x_{k+1} - x_k) \tag{2}$$

Po przekształceniu otrzymamy

$$x_{k+1} = x_k - f(x_k)/f'(x_k)$$

Korzystając z tego wzoru tworzymy ciąg punktów, który ma szansę być zbieżnym do miejsca zerowego funkcji f.

Metoda Newtona nie zawsze jest zbieżna – to zależy zarówno od funkcji f, jak i przybliżenia początkowego. Może się zdarzyć, że dla pewnego k, $f'(x_k) = 0$. Jest tak, gdy styczna w k-tym kroku jest równoległa do osi OX. Może się także zdarzyć, że $f'(x_k) \neq 0$ dla wszystkich k (we wzorze (3.1) nie wystąpi dzielenie przez zero), ale wygenerowany ciąg albo nie ma granicy albo jest rozbieżny do nieskończoności.

Twierdzenie 3.1. *Jeżeli istnieje przedział* $[a, b] \subset \mathbb{R}$*, taki, że*

1) f *jest funkcją klasy* $C^2([a, b])$ *,*

2) $f(a)\, f(b) < 0$*,*

3) f' *i* f'' *nie zmieniają znaku w* $[a, b]$*,*

4) $x_0 \in [a, b]$ *jest przybliżeniem początkowym takim, że* $f(x_0)\, f''(x_0) > 0$*,*

to metoda Newtona jest zbieżna do miejsca zerowego funkcji f*.*

Powyższe twierdzenie jest warunkiem dostatecznym. Warunek 2) zapewnia, że w przedziale [a,b] jest miejsce zerowe funkcji ciągłej.

Stosuje się również pewne modyfikacje omówionych metod pozwalające utrzymać ich rząd, a więc przyspieszające ich zbieżność. Jednym ze sposobów takiej modyfikacji jest przekształcenie otrzymanego wcześniej wzoru do postaci:

$$x_{k+1} = x_k - r \cdot f(x_k)/f'(x_k)$$

gdzie r jest krotnością pierwiastka. Zatem przy pierwiastkach dwukrotnych r = 2.

2 Opis implementacji

Zadanie zacząłem od stworzenia funkcji liczącej pochodną funkcji, ponieważ w środowisku Matlab funkcja licząca pochodną korzysta ze zmiennych symbolicznych.

```
function [dx, info] = pochodna(f, x)



dx = 0;
info = 0;
h = 0.00000000001;

A = (f(x+h) - f(x))/h;

B = (f(x-0.00000000001+h) - f(x-0.00000000001))/h;

if(A - B > 0.001)
    return
end
dx = A;
info = 1;
end
```

W funkcji tej wyznaczam liczby A oraz B i sprawdzam, czy są równe, ponieważ jak wiadomo - funkcja abs(x) nie ma pochodnej w x = 0, a współczynnik h może zbiegać do zera z lewej strony. Jeżeli współczynniki A i B są prawie równe, to pochodna w punkcie istnieje.

2.1 Metoda Newtona

Kod stosujący metodę Newtona w Matlabie:

```
eps = 0.00000000001;
info = 0;
X = 100;


lista = zeros(1,100000);

lista(1) = x0;

if f(a)*f(b) >= 0
    return
end

for k = 1:1000
    if pochodna(f, lista(k)) == 0
        X = lista(k);
        return
    end
    lista(k+1) = lista(k) - f(lista(k))/pochodna(f, lista(k));
    if abs(lista(k+1) - lista(k)) < eps
        X = lista(k+1);
        info = 1;
        return
    end
end

end
```

Ustawiam bardzo mały epsilon, wartość info (czy udało się znaleźć pierwiastek tą metodą) na zero, natomiast X na 100. Kiedy nie nastąpi ani jedna iteracja, to X taki pozostanie. Stworzyłem listę samych zer, przedstawiających kolejne wyrazy ciągu x_k. Sprawdzam warunek $f(a) * f(b) < 0$. Następnie już korzystam ze wzoru na metodę Newtona zawsze najpierw sprawdzając, czy pochodna w pewnym momencie się nie zeruje.

2.2 Zmodyfikowana metoda Newtona

Kod z rozwiązaniem znajduje się poniżej:

```
eps = 0.00000000001;
info = 0;
X = 100;


lista = zeros(1,100000);

lista(1) = x0;

if f(a)*f(b) >= 0
    return
end

for k = 1:1000
    if pochodna(f, lista(k)) == 0
        X = lista(k);
        return
    end
    lista(k+1) = lista(k) - 2*f(lista(k))/pochodna(f, lista(k));
    if abs(lista(k+1) - lista(k)) < eps
        X = lista(k+1);
        info = 1;
        return
    end
end

end
```

Kod jest prawie identyczny do poprzedniego. Różni się jedynie wyrazem 2 - w końcu stosujemy tutaj inny wzór, ale metoda iteracyjna ta sama.

3 Skrypt testujący

W tym punkcie przedstawię wyniki funkcji przeze mnie napisanych.

```
% skrypt testujący

format long;

a = @(x) x+4;
b = @(x) x^2 - 7;
c = @(x) x^2 - 4;
d = @(x) x^4 - 4*x^3 + 5*x^2 - 4*x + 4;
e = @(x) x^2*(sin(x) - cos(2*x)+1);
f = @(x) x^3*(x+sin(x^2 -1)-1) -1;
g = @(x) x - (1/3)*x^(1/3)-2;
h = @(x) x^2;
i = @(x) x^4 - 4*x^3 + 5*x^2 - 4*x + 4;
j = @(x) (x-3)^2*(x-1)*x;
k = @(x) (sin(x)-2)^3*(x+cos(x))^3*(x-2)^2;
l = @(x) (x-2)^2*(x-1);
m = @(x) (x-3)^2*(x-10);
```

Jak widać, zacząłem od zdefiniowania 13 przykładowych funkcji - 7 do metody Newtona, 6 do zmodyfikowanej metody Newtona.

```
% METODA NEWTONA ZER JEDNOKROTNYCH

[A, info] = Newton_pojedynczy(a, -3.5,-5, -3)

[B1, info] = Newton_pojedynczy(b, 2, 1,3)

[B2, info] = Newton_pojedynczy(b, -2, -3, -1)

[C1, info] = Newton_pojedynczy(c, 1, 0, 3)

[C2, info] = Newton_pojedynczy(c, -1, -3, 0)

[D, info] = Newton_pojedynczy(d, 1.5, 0, 2)

[E, info] = Newton_pojedynczy(e, 3, 2, 3.5)

[F1, info] = Newton_pojedynczy(f, -0.5, -1, 0)

[F2, info] = Newton_pojedynczy(f, 1.5, 1, 2)

[G, info] = Newton_pojedynczy(g, 3, 0, 4)
```

```
% METODA NEWTONA ZER DWUKROTNYCH

[H, info] = Newton_podwojny(h, 1, 0,2)

[I, info] = Newton_podwojny(i, 1.5, 0, 2)

[J, info] = Newton_podwojny(j, 5, 0.5, 6)

[K, info] = Newton_podwojny(k, 1, -0.9,3)

[L, info] = Newton_podwojny(l, 2.5, 0,3)

[M, info] = Newton_podwojny(m, 5, 0,11)
```

4 Wnioski

Na podstawie przeprowadzonych testów wyszło, że funkcje działają poprawnie, znajdują prawie dokładne miejsca zerowe jedno, jak i dwukrotne. Jeżeli nie są spełnione założenia Twierdzenia, to wyświetlana jest informacja, że funkcja nie została wykonana.