Metody numeryczne - Projekt II

Paulina Przybyłek
Przemysław Chojecki

9 styczeń 2020

Projekt nr 2: Wizualizacja szybkości zbieżności metody Halley'a (w dziedzinie zespolonej) zastosowanej do znalezienia zera wielomianu

$$w_n(x) = \sum_{k=0}^{n} a_k x^k.$$

Spis treści

1 Opis metody

1.1 Metoda Halley'a

Metoda Halley'a to algorytm iteracyjny wyznaczania przybliżonej wartości pierwiastka funkcji $f(x)$ jednej zmiennej $x = x_k + y_j i$ w zadanym przedziale $x_k \in [a, b]$ oraz $y_j \in [c, d]$. Kolejne przybliżenia oblicza się w sposób rekurencyjny.

Niech $f : \mathbb{C} \to \mathbb{C}$ będzie klasy $C^2(\mathbb{C})$ oraz niech $x_0 \in \mathbb{C}$ będzie danym przybliżeniem początkowym.
Wykorzystując rozwinięcie funkcji $f(x)$ w szereg Taylora w otoczeniu punktu x_k otrzymujemy przybliżenie $p(x)$ funkcji $f(x)$:

$$p(x) = f(x_k) + f'(x_k)(x - x_k) + \frac{1}{2}f''(x_k)(x - x_k)^2.$$

Następnie przyjmujemy:

$$0 = f(x_k) + f'(x_k)(x_{k+1} - x_k) + \frac{1}{2}f''(x_k)(x_{k+1} - x_k)^2.$$

Po wykonaniu odpowiednich przekształceń otrzymujemy:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k) + \frac{1}{2}f''(x_k)(x_{k+1} - x_k)}.$$

Wartość x_{k+1} po prawej stronie przybliżamy metodą Newtona otrzymując wzrór określający metodę Halley'a:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k) + f''(x_k)\frac{f(x_k)}{2f'(x_k)}} \text{ dla } k = 0, 1 \ldots$$

Więc ostatecznie uzyskujemy poniższy wzór i to właśnie on jest wykorzystywany do stworzenia algorytmu metody Halley'a.

$$x_{k+1} = x_k - \frac{2f(x_k)f'(x_k)}{2(f'(x_k))^2 + f''(x_k f(x_k)} \text{ dla } k = 0, 1 \ldots$$

Warunkiem zakończenia obliczeń może być jeden z poniższych warunków stopu:

1. $|x_{k+1} - x_k| < \epsilon$

2. $|x_{k+1} - x_k| < \epsilon|x_k|$

3. $|x_{k+1} - x_k| < \epsilon|x_k| + \delta$

4. $|f(x_k)| < \epsilon$ (w projekcie wybrany został ten warunek stopu)

gdzie ϵ i δ określają dokładność.

Idea Metody Halley'a:
Kolejne przybliżenie x_{k+1} jest miejscem zerowym hiperboli przybliżającej funkcję $f(x)$ w punkcie x_k.

1.2 Zbieżność metody Halley'a

$$x_k \to x \text{ przy } k \to \inf$$

DEFINICJA Wykładnikiem zbieżności metody iteracyjnej nazywamy największą liczbę $p \geq 1$ taką, że:

$$|e_{k+1}| \leq C|e_k|^p$$

gdzie $e_k = x_k - x$ jest błędem w k-tym kroku, a C pewną stałą (nieujemną).
Dla metody Halley'a $p = 3$, czyli metoda ma zbieżność sześcienną.

Wizualizacja szybkości zbieżności W projekcie wizualizacja przedstawiona jest na podstawie liczby iteracji wykonania metody Halley'a do momentu znalezienia pierwiastka funkcji $f(x) = 0$ w dziedzinie zespolonej, który spełnił warunek stopu.

2 Opis funkcjonalności metody w Matlabie

Opisana metoda została ujęta w funkcji **MetodaHalleya.m**. Jednak do rozwiązania zadania, należało posiadać dodatkowe funkcje, które ułatwiły wizualizację szybkości zbieżnośći metody. Stworzono odpowiednie funkcje do tworzenia macierzy punktów początkowych przybliżeń należących do dziedziny zespolonej (**MacierzA.m**), do wykorzystywania metody Halleya dla danej macierzy (**HalleyMatrix.m**) oraz zamiany macierzy pierwiastków funkcji na macierz liczb $k \in \{-1, 1, 2, \dots\}$, gdzie -1 oznacza NaN, czyli brak pierwiastka w zadanym przedziale i dla danego punktu początkowego (**zamianaCnaD.m**).

2.1 Cel zadania i sposób jego rozwiązania

Rozwiązaniem zadania jest przedstawienie szybkości zbieżności metody - czyli wygenerowanie wykresów w Matlabie na podstawie liczby iteracji do uzyskania pierwiastka. Dodatkowo wykonano też wykresy ukazujące zbieżność do danych pierwiastków w zależności od początkowych przybliżeń. Szczegóły funkcji opisane zostały w poniższym akapicie.

2.2 Funkcje zawarte w programie

[x, k, w_x] = MetodaHalleya(p, x0, tol, max_iter)

Funkcja znajduje zero wielomianu metodą Halley'a. Przyjmuje od użytkownika:

- p - czyli współczynniki wielomianu (naszej funkcji f)
- x_0 - początkowe przybliżenie
- *tol* - dokładność obliczeń (stosowany ϵ do warunku stopu)
- max_{iter} - maksymalna liczba iteracji

Zwracane jest odpowiednio przybliżone zero wielomianu, liczba iteracji do momentu znalezienia pierwiastka oraz wartość wielomianu dla obliczonego pierwiastka.

UWAGA: Przy wizualizacji iteracje powyżej 30 są kolorowane na czarno, dlatego jeśli nie zostanie znaleziony pierwiastek zwracany jest NaN oraz $k = 31$.

```
1 function [x, k, w_x] = MetodaHalleya(p, x0, tol,
    max_iter)
2
3 k = 0;
4 dx = tol + 1;
5 xpocz = x0;
6
7 dw = polyder(p);
8 ddw = polyder(dw);
9
```

```
10 while abs(dx) > tol && k <= max_iter
11     w = polyval(p,x0);
12     if abs(w) <= tol
13         x = x0;
14         w_x = w;
15         return
16     end
17     dzielnik = 2*polyval(dw,x0)^2 - (w*polyval(ddw,
           x0));
18
19     if dzielnik == 0
20         disp('Dzielenie przez zero!');
21         return
22     end
23
24     dx = (2*w*polyval(dw, x0))/dzielnik;
25     x1 = x0 - dx;
26     k = k + 1;
27     x0 = x1;
28 end
29
30 fprintf('Nie znaleziono rozwiazania w %d iteracjach,
       zaczynajac od %d z wymagana precyzja wynoszacej:
     %d \n', max_iter, xpocz, tol);
31 k = 31;
32 x = NaN;
33 end
```

A = MacierzA(a,b,c,d,n,m)

Funkcja zwraca macierz $A((n+1) \times (m+1))$, która na miejscu $a_{k,j}$ zawiera punkt z dziedziny zespolonej $x = x_k + y_j i$ w zadanym przedziale $x_k \in [a, b]$ oraz $y_j \in [c, d]$, których krańce przedziałów podaje użytkownik jako parametry funkcji.
Dodatkowo podane n oraz m oznaczają granice wyznaczania wartości x_k

oraz y_j. Dany podział uzyskiwany jest poprzez wykorzystanie wzorów:

$$x_k = a + kh_1 \text{ gdzie } h_1 = \frac{b-a}{n}, \text{ dla } k = 0, 1, \ldots, n$$

$$y_j = c + jh_2 \text{ gdzie } h_2 = \frac{d-c}{m}, \text{ dla } j = 0, 1, \ldots, m$$

```
function A = MacierzA(a,b,c,d,n,m)

h1 = (b - a)/n;
h2 = (d - c)/m;
A = zeros(n+1, m+1);
x = zeros(1, n+1);
y = zeros(1, m+1);

for k=1:n+1
    x(1,k) = a + (k-1) * h1;
end
for j=1:m+1
    y(1,j) = c + (j-1) * h2;
end
for k=1:n+1
    for j=1:m+1
        A(k,j) = x(1,k) + y(1, j) * i;
    end
end
end
```

[B,C,D] = HalleyMatrix(p, A, tol, max_iter)

Funkcja jest tak jakby połączeniem dwóch powyższych funkcji. Przyjmowane argumenty są prawie takie jak w **MetodaHalleya.m**, z tą różnicą, że zamiast początkowego przybliżenia x_0 użytkownik podaje odpowiednią macierz A, czyli taką jaką zwraca funkcja **MacierzA.m**. Wtedy algorytm oblicza metodą Halley'a pierwiastek dla każdej komórki z A, tzn. a_{kj} jest początkowym przybliżeniem.
Po wywołaniu funkcji uzyskujemy trzy macierze:

- B, kóra zawiera w komórkach odpowiadających tym z A ilość iteracji do uzyskania pierwiastka

- C, która zawiera w komórkach odpowiadających tym z A pierwiastek, do którego jest zbieżne
- D - macierz liczb $\{-1, 1, 2, \dots\}$, które przypisują daną liczbę jednemu pierwiastkowi (uzyskujemy tę macierz dzięki funkcji **zamianaCnaD.m**)

```
function [B,C,D] = HalleyMatrix(p, A, tol, max_iter)

[n,m] = size(A);
B = zeros(n,m);
C = zeros(n,m);
for i=1:n
    for j=1:m
        [x,k] = MetodaHalleya(p, A(i,j), tol,
            max_iter);
        B(i,j) = k;
        C(i,j) = x;
    end
end
D = zamianaCnaD(C);
end
```

D = zamianaCnaD(C)

Funkcja przyjmuje macierz miejsc zerowych C oraz zwraca macierz liczb całkowitych D, gdzie każda liczba przypisana jest do jednego pierwiastka z macierzy C. Jeśli pierwiastek nie istnieje, czyli jest zapisany jako NaN, to przypisana jest mu liczba -1.

```
function [D] = zamianaCnaD(C)

[n,m] = size(C);
D = zeros(n,m);
for i=1:n
    for j=1:m
        if isnan(real(C(i,j))) || isnan(imag(C(i,j))
            )
            D(i,j) = -1;
```

```
9           end
10      end
11  end
12  p = 1;
13  tol = 0.1;
14
15  for i=1:n
16      for j=1:m
17          if D(i,j) == 0
18              D(i,j) = p;
19              for k=1:n
20                  for l=1:m
21                      if isnan(real(C(k,l))) || isnan(
                            imag(C(k,l)))
22                          continue
23                      end
24                      if abs(C(i,j) - C(k,l)) <= tol
25                          D(k,l) = p;
26                      end
27                  end
28              end
29          p = p + 1;
30          end
31      end
32  end
33  end
```

3 Ciekawe przykłady i wizualizacja wyników

3.1 Wizualizacja szybkości zbieżności metody

3.2 Zbieżność przedstawiona za pomocą pierwiastków wielomianu

4 Analiza wyników