Metoda Halley'a wyznaczania miejsc zerowych wielomianu zapisanego w bazie wielomianów Czebyszewa I-rodzaju

Maciej Chylak, grupa 1

26 listopada 2020r.

Spis treści

1 Temat

1.1 Treść zadania

Metoda Halley'a wyznaczania zer wielomianu

$$w_n(x) = \sum_{k=0}^{n} a_k T_k(x) T_{n-k}(x)$$

gdzie $T_0, T_1, ..., T_n$ są wielomianami Czebyszewa I-rodzaju.
Uwaga. Nie należy sprowadzać wielomianu w_n do postaci naturalnej! Do obliczania wartości wielomianu w_n oraz jego pochodnych należy wykorzystać związek rekurencyjny spełniany przez wielomiany Czebyszewa.

Głównym tematem powyższego zadania jest znajdywanie miejsc zerowych wielomianu zapisanego w bazie wielomianów Czebyszewa I-stopnia przy użyciu metody Halley'a. Wielomiany Czebyszewa są to wielomiany spełniające poniższą zależność:

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_n(x) &= 2xT_{n-1}(x) - T_{n-2}(x) \qquad \text{dla} \quad \text{n} \geq 2 \end{aligned}$$

Wzór jawny na n-ty wielomian Czebyszewa:

$$T_n(x) = \frac{(x + \sqrt{x^2-1}^n) + (x - \sqrt{x^2-1}^n)}{2}$$

1.2 Główne zastosowanie wielomianów Czebyszewa

W trakcie interpolowania wielomianu bardzo często zamiast równoodległych wezłów stosuję się węzły Czebyszewa, czyli pierwiastki wielomianu Czebyszewa. Dzięki temu jesteśmy w stanie uniknąć zjawiska Rungego, czyli dużych oscylacji wielomianu interpolacyjnego na krańcach przedziału, gdyż miejsca zerowe wielomianów Czebyszewa zagęszczają się na krańcach przedziałów.

1.3 Metoda Halley'a

Dla funkcji $f : \mathbb{R} \to \mathbb{R}$ będącej klasy $C^2(\mathbb{R})$ oraz x_0 będącym przybliżeniem początkowym miejsca zerowego funkcji, kolejne przybliżenie będziemy wyznaczali przy pomocy rozwinięcia funkcji $f(x)$ w szereg Taylora w otoczeniu punktu x_k gdzie k oznacza k-te przybliżenie miejsca zerowego funkcji f a x oznacza przybliżone miejsce zerowe funkcji f:

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

Przyjmujemy, że $f(x) = 0$, gdyż x jest naszym hipotetycznym miejscem zerowym:

$$0 = f(x_k) + \frac{f'(x_k)(x - x_k)^1}{1!} + \frac{f''(x_k)(x - x_k)^2}{2!}$$

Następnie przyjmujemy, że $x = x_{k+1}$ oznaczające następne przybliżenie naszego miejsca zerowego. Po odpowiednich przekształceniach otrzymujemy wzór na kolejne przybliżenie miejsca zerowego funkcji x_{k+1}:

$$x_{k+1} = x_k - \frac{2f(x_k)f'(x_k)}{2(f'(x_k)^2) - f(x_k)f''(x_k)} \qquad \text{dla} \quad \mathrm{k} \in \mathbb{N} \cap \{0\}$$

Przyjętym przeze mnie warunkiem stopu algorytmu będzię spełnienie następującej nierówności dla ustalonego d:

$$|x_{k+1} - x_k| < d$$

Gdy dla pewnego $x_n \quad f'(x_n) = 0$ to metoda Halleya nie będzie zbieżna, gdyż następuję dzielenie przez 0.

2 Program obliczeniowy

2.1 Opis funkcji

Program obliczeniowy składa się z kilku funcji pomocniczych oraz jednej funkcji głównej, korzystającej z pozostałych.
Funkcje pomocnicze składają się z następujących funkcji:

- w_czeb_wart(n, x) - pozwalająca obliczyć wartość wielomianu Czebyszewa n-tego stopnia w punkcie x;
- w_czeb_poch_wart(n, x) - pozwalająca obliczyć wartość pierwszej pochodnej wielomianu Czebyszewa n-tego stopnia w punkcie x;
- w_czeb_poch2_wart(n, x) - pozwalająca obliczyć wartość drugiej pochodnej wielomianu Czebyszewa n-tego stopnia w punkcie x;
- w_wart(a, x) - pozwalająca obliczyć wartość wielomianu zapisanego w bazie wielomianów Czebyszewa dla wektora współczynników a w punkcie x;
- w_poch_wart(a, x) - pozwalająca obliczyć wartość pierwszej pochodnej wielomianu zapisanego w bazie wielomianów Czebyszewa dla wektora współczynników a w punkcie x;
- w_poch2_wart(a, x) - pozwalająca obliczyć wartość drugiej pochodnej wielomianu zapisanego w bazie wielomianów Czebyszewa dla wektora współczynników a w punkcie x.

A funkcje główne z następującej:

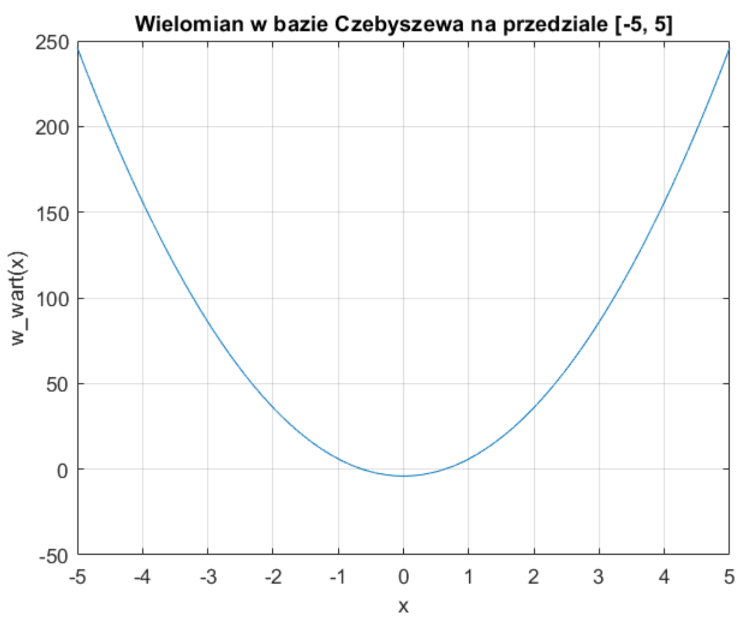
- metodaHalleya(x, a, d, N) - pozwalająca wyznaczyć miejsce zerowe wielomianu zapisanego w bazie wielomianów Czebyszewa dla x - oznaczającego przybliżenie początkowe wyznaczanego miejsca zerowego, a - wektora współczynników, d - parametru określającego dokładność obliczanego miejsca zerowego oraz N - maksymalnej liczby iteracji programu, po której przekroczeniu program stwierdza, że algorytm nie zbiega do miejsca zerowego przy ustalonym d w N krokach.

2.2 Sposób obsługi programu

Program obsługujemy poprzez wywołaniu funkcji metodaHalleya(x, a, d, N) dla ustalonych przez nas argumentów. W wywołania funkcji otrzymamy najbliż przybliżone miejsce zerowe wielomianu zapisanego w bazie wielomianów Czebyszewa I-stopnia lub, w przypadku przekroczenia limitu iteracji, komunikat z niepowodzeniem odnalezienia miejsca zerowego przy danym przybliżeniu w N krokach, natomiast w przypadku, gdy wartości pierwszej pochodnej dla pewnego x_n będzie równa 0, to zostanie wyświetlony komunikat, że algorytm nie jest zbieżny.

2.3 Przykład obliczeniowy nr 1

$x = 0, \quad a = [1\ 2\ 3] \quad d = 0.001 \quad N = 40.$
Wykres takiego wielomianu prezentuje się następująco:



Wywołanie funkcji zwraca nam następującą rzecz:

```
>> metodaHalleya(0, [1 2 3], 0.001, 40)
Algorytm nie jest zbieżny

ans =

   NaN
```

Gdyż wartość pochodnej w punkcie x_n dla n = 0 jest równa 0:

```
>> w_poch_wart(a, 0)

ans =

     0
```

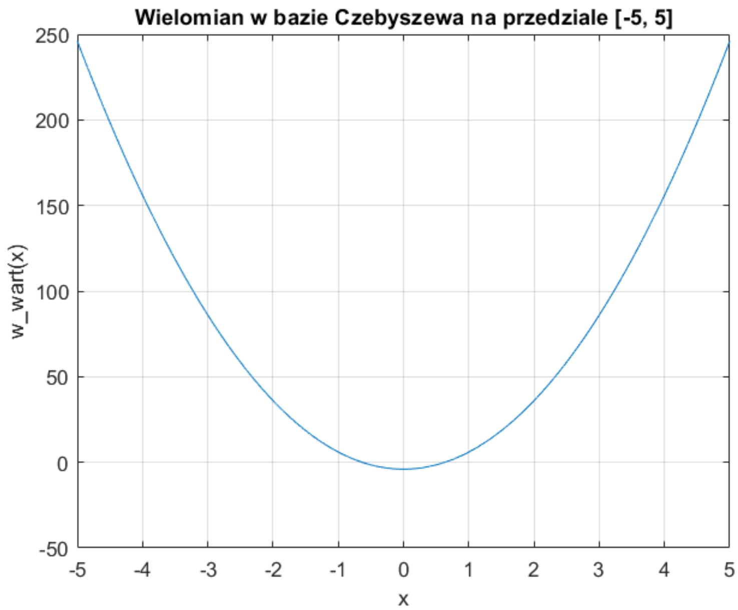
Wynik przy użyciu funkcji fzero:

```
>> fzero(@(x) w_wart([1 2 3], x), 0)

ans =

   -0.6325
```

Oczywiście fzero używa innej metody do wyznaczanie miejsc zerowych, więc nie dziwi nas fakt, że wynik uzyskany przy pomocy funkcji fzero jest dobry, a funkcja metodaHalleya nie wyznacza miejsca zerowego funkcji

2.4 Przykład obliczeniowy nr 2

$x = 0.001, \quad a = [1\,2\,3] \quad d = 0.001 \quad N = 10.$
Wykres takiego wielomianu prezentuje się następująco:



Wywołanie funkcji zwraca nam następującą rzecz:

```
>> metodaHalleya(0.001, [1 2 3], 0.001, 10)

ans =

    0.6325
```

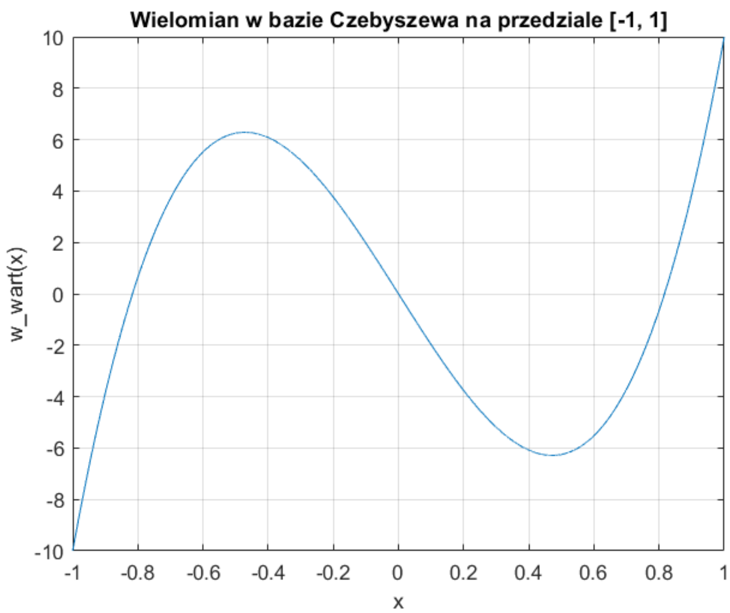
Wynik przy użyciu funkcji fzero:

```
>> fzero(@(x) w_wart([1 2 3], x), 0.001)

ans =

   -0.6325
```

Wyniki dla obu funkcji są różne; funkcja fzero znalazła inne miejsce zerowe, lecz spoglądając na wykres jesteśmy w stanie stwierdzić, że oba wyniki są najprawdopodobniej poprawne. Utwierdza nas w przekonaniu również to, że miejsca zerowe są symetryczne wzgledem osi OX, ponieważ obliczamy miejsca zerowe dla funkcji kwadratowej, dla której x wierzchołka jest równy 0.

2.5 Przykład obliczeniowy nr 3

$x = 1, \quad a = [1\,2\,3\,4] \quad d = 0.00001 \quad N = 10.$
Wykres takiego wielomianu prezentuje się następująco:



Wywołanie funkcji zwraca nam następującą rzecz:

```
>> metodaHalleya(1, [1 2 3 4], 0.00001, 10)

ans =

    0.8165
```

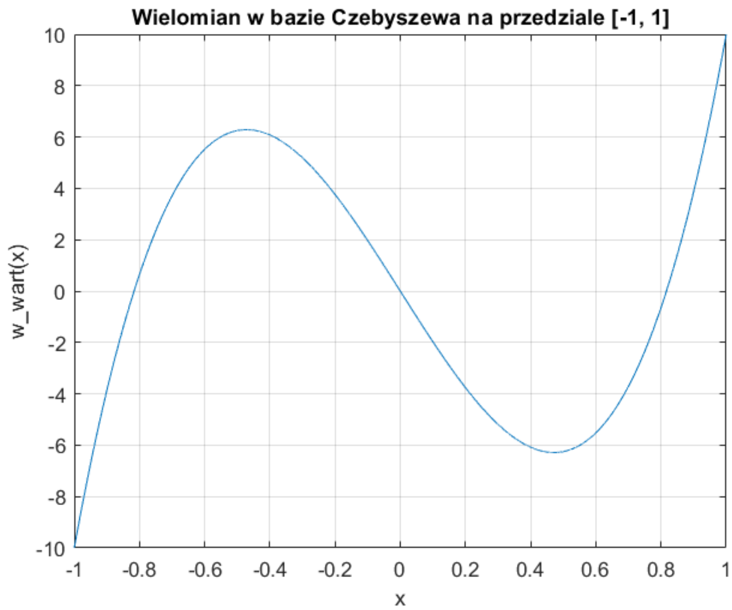
Wynik przy użyciu funkcji fzero:

```
>> fzero(@(x) w_wart([1 2 3 4], x), 1)

ans =

    0.8165
```

Wyniki dla obu funkcji są tożsame z dokładnością do d.

2.6 Przykład obliczeniowy nr 4

$x = 0.05$, $a = [1\,2\,3\,4]$ $d = 0.00001$ $N = 10$.
Wykres takiego wielomianu prezentuje się następująco:



Wywołanie funkcji zwraca nam następującą rzecz:

```
>> metodaHalleya(0.05, [1 2 3 4], 0.00001, 10)

ans =

     0
```

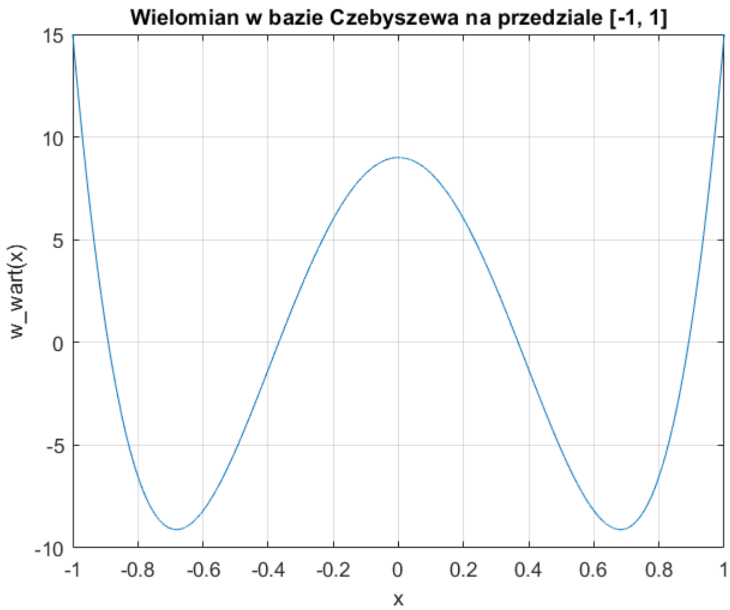
Wynik przy użyciu funkcji fzero:

```
>> fzero(@(x) w_wart([1 2 3 4], x), 0.05)

ans =

  -6.3109e-30
```

Wyniki dla obu funkcji są tożsame z dokładnością do d.

2.7 Przykład obliczeniowy nr 5

$x = 0, \quad a = [1\,2\,3\,4\,5] \quad d = 0.00001 \quad N = 10.$
Wykres takiego wielomianu prezentuje się następująco:



Wywołanie funkcji zwraca nam następującą rzecz:

```
>> metodaHalleya(0, [1 2 3 4 5], 0.00001, 10)
Algorytm nie jest zbieżny

ans =

   NaN
```

Gdyż wartość pochodnej w punkcie x_n dla n = 0 jest równa 0:

```
>> w_poch_wart(0, [1 2 3 4 5])

ans =

     0
```

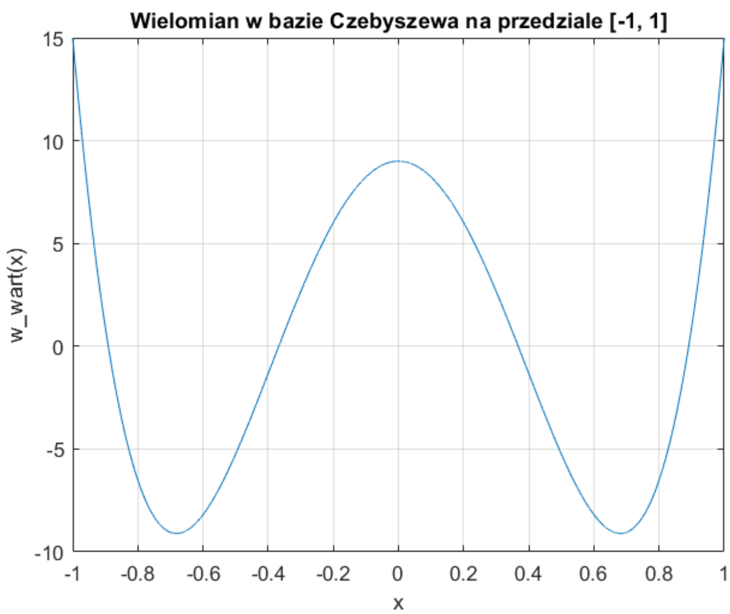
Wynik przy użyciu funkcji fzero:

```
>> fzero(@(x) w_wart([1 2 3], x), 0)

ans =

   -0.6325
```

Oczywiście fzero używa innej metody do wyznaczanie miejsc zerowych, więc nie dziwi nas fakt, że wynik uzyskany przy pomocy funkcji fzero jest dobry, a funkcja metodaHalleya nie wyznacza miejsca zerowego funkcji

2.8 Przykład obliczeniowy nr 6

$x = -0.4, \quad a = [1\,2\,3\,4\,5] \quad d = 0.00001 \quad N = 1.$
Wykres takiego wielomianu prezentuje się następująco:



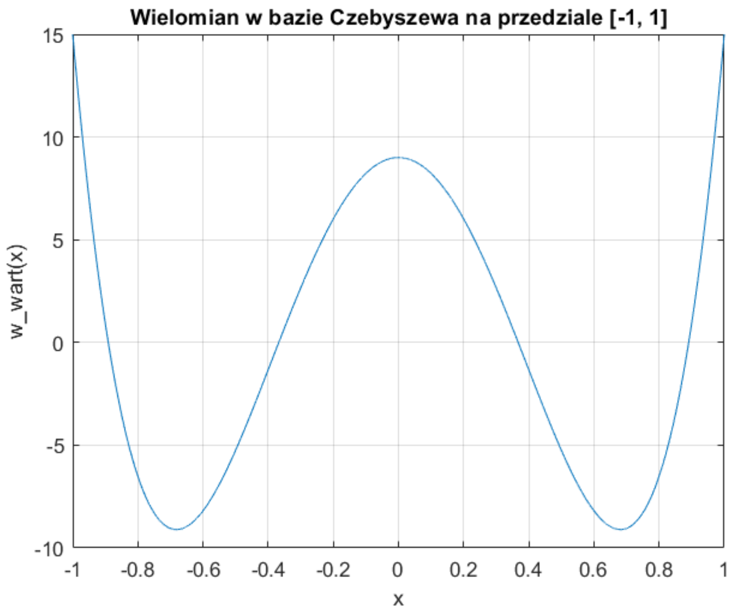
Wywołanie funkcji zwraca nam następującą rzecz:

```
>> metodaHalleya(-0.4, [1 2 3 4 5], 0.00001, 1)
Program nie zbiega do miejsca zerowego z podaną dokładnością w ustalonej liczbie kroków

ans =

   NaN
```

Wykonaliśmy zbyt mało iteracji. Dla nieco większego N algorytm powinien zbiec do miejsca położonego w okolicy punktu -0.3674 (Zostało to pokazane, w przykładzie obliczeniowym nr 7)

2.9 Przykład obliczeniowy nr 7

$x = -0.4, \quad a = [1\,2\,3\,4\,5] \quad d = 0.00001 \quad N = 10.$
Wykres takiego wielomianu prezentuje się następująco:



Wywołanie funkcji zwraca nam następującą rzecz:

```
>> metodaHalleya(-0.4, [1 2 3 4 5], 0.00001, 10)

ans =

  -0.3674
```

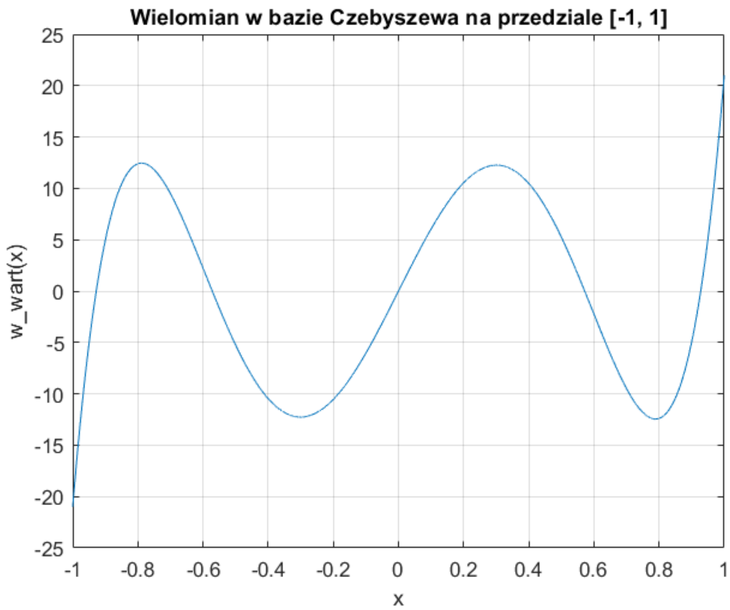
Wynik przy użyciu funkcji fzero:

```
>> fzero(@(x) w_wart([1 2 3 4 5], x), -0.4)

ans =

   -0.3674
```

Wyniki dla obu funkcji są tożsame z dokładnością do d.

2.10 Przykład obliczeniowy nr 8

$x = -0.4$, $a = [1\,2\,3\,4\,5\,6]$ $d = 0.00001$ $N = 10$.
Wykres takiego wielomianu prezentuje się następująco:



Wywołanie funkcji zwraca nam następującą rzecz:

```
>> metodaHalleya(-0.4, [1 2 3 4 5 6], 0.00001, 10)

ans =

   -0.5716
```

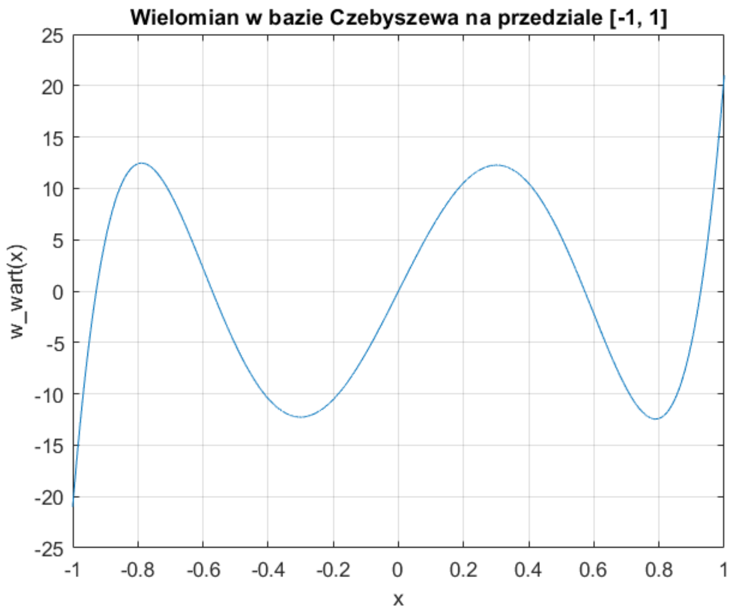
Wynik przy użyciu funkcji fzero:

```
>> fzero(@(x) w_wart([1 2 3 4 5 6], x), -0.4)

ans =

   -0.5716
```

Wyniki dla obu funkcji są tożsame z dokładnością do d.

2.11 Przykład obliczeniowy nr 9

$x = -0.25, \quad a = [1\,2\,3\,4\,5\,6] \quad d = 0.00001 \quad N = 10.$
Wykres takiego wielomianu prezentuje się następująco:



Wywołanie funkcji zwraca nam następującą rzecz:

```
>> metodaHalleya(-0.25, [1 2 3 4 5 6], 0.00001, 10)

ans =

     0
```

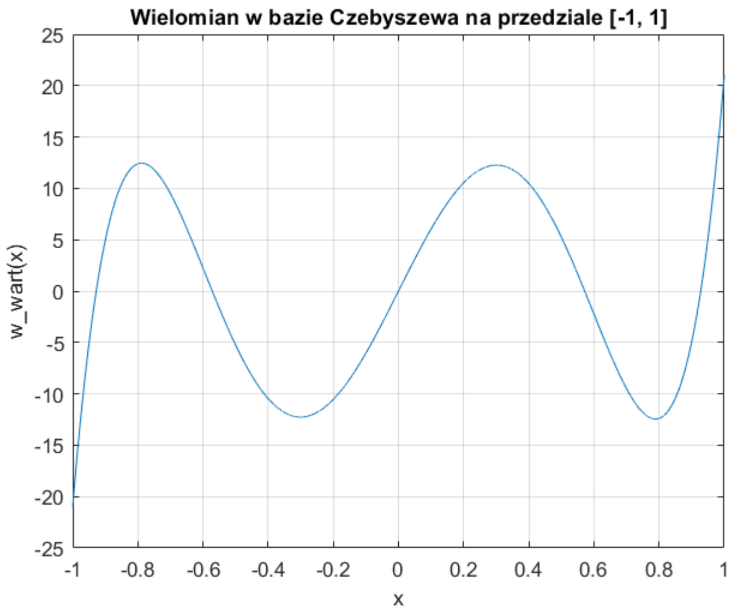
Wynik przy użyciu funkcji fzero:

```
>> fzero(@(x) w_wart([1 2 3 4 5 6], x), -0.25)

ans =

   4.9211e-17
```

Wyniki dla obu funkcji są tożsame z dokładnością do d.

2.12 Przykład obliczeniowy nr 10

$x = 0.4$, $\quad a = [1\,2\,3\,4\,5\,6] \quad d = 0.00001 \quad N = 10$.
Wykres takiego wielomianu prezentuje się następująco:



Wywołanie funkcji zwraca nam następującą rzecz:

```
>> metodaHalleya(0.4, [1 2 3 4 5 6], 0.00001, 10)

ans =

    0.5716
```

Wynik przy użyciu funkcji fzero:

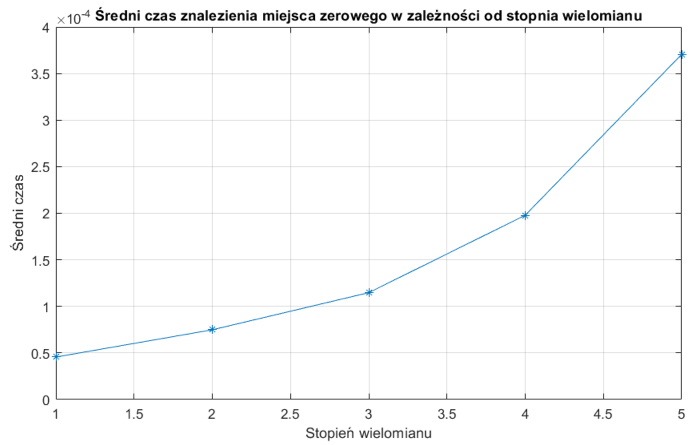
```
>> fzero(@(x) w_wart([1 2 3 4 5 6], x), 0.4)

ans =

    0.5716
```

Wyniki dla obu funkcji są tożsame z dokładnością do d.

2.13 Analiza prędkości działania algorytmu w zależności od stopnia wielomianu

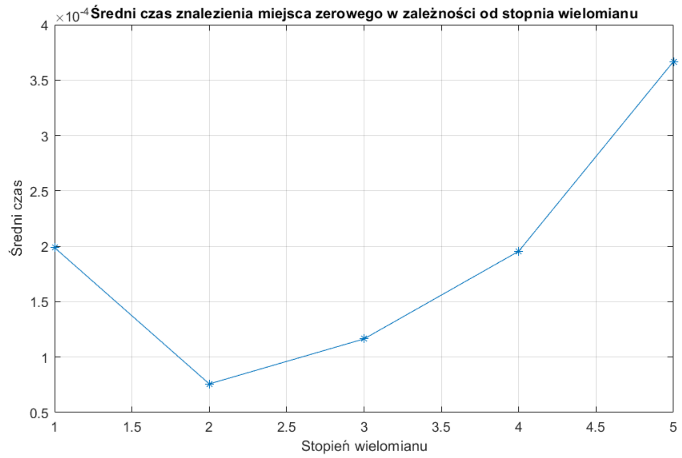
- Średni czas znalezienia miejsca zerowego 5 różnych wektorów a:
  $a = [1, 1]$, $a = [1, 1, 1]$, $a = [1, 1, 1, 1]$, $a = [1, 1, 1, 1, 1]$, $a = [1, 1, 1, 1, 1, 1]$
  dla 10 różnych x początkowych: 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9

  Pomiar nr 1:



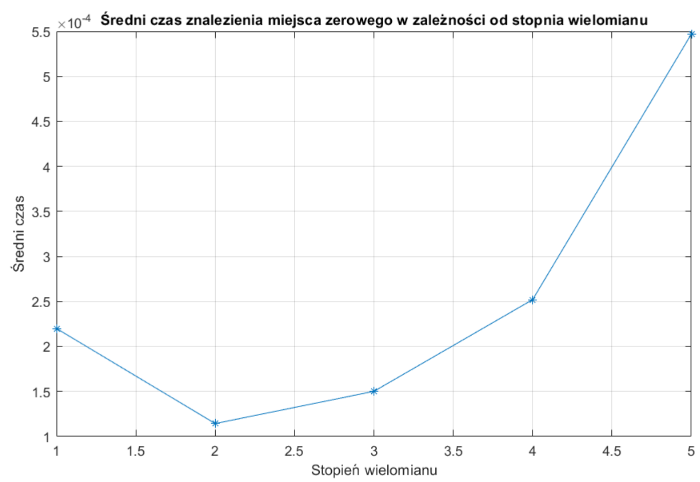
Pomiar nr 2:

Pomiar nr 3:

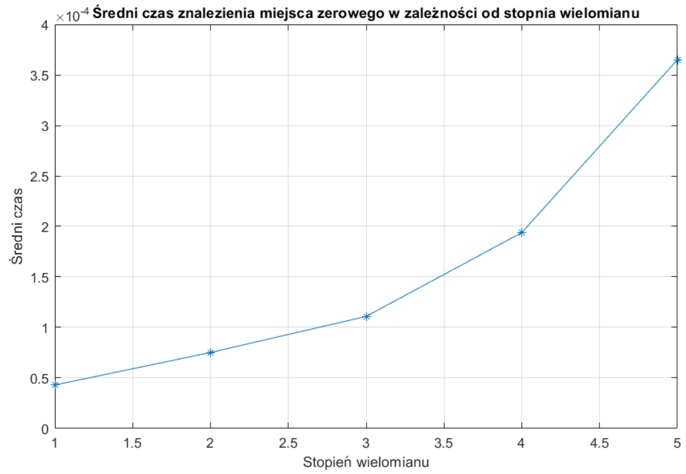


- Średni czas znalezienia miejsca zerowego 5 różnych wektorów a:
  $a = [1, 2]$, $a = [1, 2, 3]$, $a = [1, 2, 3, 4]$, $a = [1, 2, 3, 4, 5]$, $a = [1, 2, 3, 4, 5, 6]$
  dla 10 różnych x początkowych: 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9

  Pomiar nr 1:



Pomiar nr 2:



Pomiar nr 3:

Jak widać na powyższych wykresach średni czas znalezienia miejsca zerowego rośnie wraz ze wzrostem stopnia wielomianu, jednak czasami wyznaczanie miejsca zerowego dla wielomianu stopnia 1 bywa porównywalnie długie jak w przypadku wielomianu stopnia 3, 4.

3 Bibliografia

- `https://www.mathworks.com/help/matlab/`
- `http://www.mini.pw.edu.pl/(tylda)iwrobel/MADMN\_zima\_2020-21/mn\_pliki/Zapiski\_numeryczne\_MAD.pdf`
- `https://pl.wikipedia.org/wiki/Wielomiany\_Czebyszewa`