Karol Kosoń

Zadanie O6

Zadanie O6 (obowiązkowe)

Stosując metodę Laguerre'a wraz ze strategią obniżania stopnia wielomianu i wygładzania proszę znaleźć wszystkie rozwiązania następujących równań

$$243z^7 - 486z^6 + 783z^5 - 990z^4 + 558z^3 - 28z^2 - 72z + 16 = 0,$$

$$z^{10} + z^9 + 3z^8 + 2z^7 - z^6 - 3z^5 - 11z^4 - 8z^3 - 12z^2 - 4z - 4 = 0,$$

$$z^4 + iz^3 - z^2 - iz + 1 = 0.$$

Zadanie polegało na znalezieniu miejsc zerowych powyższych wielomianów. Zgodnie z Podstawowym Twierdzeniem Algebry, wielomian posiada tyle pierwiastków jakiego jest stopnia.

ALGORYTM:

1. Musimy obrać sobie jakiś punkt początkowy
2. Wyliczamy z' metodą Laguerra

$$z_{i+1} = z_i - \frac{n\,P_n(z_i)}{P_n'(z_i) \pm \sqrt{(n-1)\left((n-1)\,[P_n'(z_i)]^2 - n\,P_n(z_i)\,P_n''(z_i)\right)}}\,, \tag{11}$$

   a.
   b. W pętli liczymy po przez algorytm Hornera wartości w danym wielomianie, jego pierwszej pochodnej, jego drugiej pochodnej.
   c. Znane wartości wstawiamy do równania
   d. Ważne też jest żeby wybrać odpowiedni znak mianownika (prosty if wystarcza)
3. Wygładzamy miejsce zerowe po przez użycie na nim ponownie metody Laguerra dla niezmienionego wielomianu.
4. Zmniejszamy stopień wielomianu przez deflację:

$$\begin{bmatrix} 1 & & & & & \\ -z_0 & 1 & & & & \\ & -z_0 & 1 & & & \\ & & \ddots & \ddots & & \\ & & & -z_0 & 1 & \\ & & & & -z_0 & 1 \end{bmatrix} \begin{bmatrix} b_{n-1} \\ b_{n-2} \\ b_{n-3} \\ \vdots \\ b_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} a_n \\ a_{n-1} \\ a_{n-2} \\ \vdots \\ a_2 \\ a_1 \end{bmatrix}$$

   Szybkość takiego działania jest bardzo duża, wykonujemy to w O(n);
5. Powtarzamy całą operację w pętli aż do 1 stopnia wielomianu.

Wyniki:

$$243z^7 - 486z^6 + 783z^5 - 990z^4 + 558z^3 - 28z^2 - 72z + 16 = 0,$$

(0.667027,0)
(0.666486,-0.000312597)
(0.666486,0.000312597)
(0.333333,0)

(0,-1.41421)
(-0.333333,0)
(0,1.41421)

```
In[5]:= y = 243 z^7 - 486 z^6 + 783 z^5 - 990 z^4 + 558 z^3 - 28 z^2 - 72 z + 16
Out[5]= 16 - 72 z - 28 z^2 + 558 z^3 - 990 z^4 + 783 z^5 - 486 z^6 + 243 z^7

In[7]:= NSolve[y, z]
        [rozwiąż numerycznie
Out[7]= {{z -> -0.333333}, {z -> 0. - 1.41421 i}, {z -> 0. + 1.41421 i}, {z -> 0.333333}, {z -> 0.666667}, {z -> 0.666667}, {z -> 0.666667}}
```

$$z^{10} + z^9 + 3z^8 + 2z^7 - z^6 - 3z^5 - 11z^4 - 8z^3 - 12z^2 - 4z - 4 = 0,$$

(-0.5,0.866025)
(-0.5,-0.866025)
(0,1.41421)
(-1.41421,0)
(0,-1.41421)
(0,1.00005)
(0,-1.00004)
(0,-0.999961)
(0,0.99995)
(1.41421,0)

```
In[8]:= g = z^10 + z^9 + 3 z^8 + 2 z^7 - z^6 - 3 z^5 - 11 z^4 - 8 z^3 - 12 z^2 - 4 z - 4
Out[8]= -4 - 4 z - 12 z^2 - 8 z^3 - 11 z^4 - 3 z^5 - z^6 + 2 z^7 + 3 z^8 + z^9 + z^10

In[9]:= NSolve[g, z]
        [rozwiąż numerycznie
Out[9]= {{z -> -1.41421}, {z -> -0.5 + 0.866025 i}, {z -> -0.5 - 0.866025 i}, {z -> 0. + 1. i}, {z -> 0. + 1. i}, {z -> 0. - 1. i}, {z -> 0. - 1. i}, {z -> 0. - 1.41421 i}, {z -> 0. + 1.41421 i}, {z -> 1.41421}}
```

$$z^4 + iz^3 - z^2 - iz + 1 = 0.$$

(-0.951057,0.309017)
(-0.587785,-0.809017)
(0.587785,-0.809017)
(0.951057,0.309017)

```
In[32]:= k = z^4 + z^3 i - z^2 - iz + 1
Out[32]= 1 - z^2 + i z^3 + z^4 - iz

In[33]:= NSolve[k, z]
         [rozwiąż numerycznie
Out[33]= {{z -> (0. - 0.25 i) + 0.5 Sqrt[(0.416667 + (0.419974 (13. - 12. iz))/(43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) + 0.264567 (43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3))] -
    0.5 Sqrt[(0.833333 - (0.419974 (13. - 12. iz))/(43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) - 0.264567 (43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) -
      (0. + 0.75 i)/(Sqrt[(0.416667 + (0.419974 (13. - 12. iz))/(43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) + 0.264567 (43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3))]))]},
  {z -> (0. - 0.25 i) + 0.5 Sqrt[(0.416667 + (0.419974 (13. - 12. iz))/(43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) + 0.264567 (43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3))] + 0.5 Sqrt[(0.833333 - (0.419974 (13. - 12. iz))/(43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) -
    0.264567 (43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) - (0. + 0.75 i)/(Sqrt[(0.416667 + (0.419974 (13. - 12. iz))/(43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) + 0.264567 (43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3))]))]},
  {z -> (0. - 0.25 i) - 0.5 Sqrt[(0.416667 + (0.419974 (13. - 12. iz))/(43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) + 0.264567 (43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3))] - 0.5 Sqrt[(0.833333 - (0.419974 (13. - 12. iz))/(43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) -
    0.264567 (43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) + (0. + 0.75 i)/(Sqrt[(0.416667 + (0.419974 (13. - 12. iz))/(43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) + 0.264567 (43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3))]))]},
  {z -> (0. - 0.25 i) - 0.5 Sqrt[(0.416667 + (0.419974 (13. - 12. iz))/(43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) + 0.264567 (43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3))] + 0.5 Sqrt[(0.833333 - (0.419974 (13. - 12. iz))/(43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) -
    0.264567 (43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) + (0. + 0.75 i)/(Sqrt[(0.416667 + (0.419974 (13. - 12. iz))/(43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3) + 0.264567 (43. - 45. iz + 5.19615 Sqrt[-257. + 758. iz - 757. iz^2 + 256. iz^3])^(1/3))]))]}}
```

Niestety do tego ostatniego Mathematica podaje powyższy wynik. Nie potrafię go zinterpretować.

KOD:

```
#include <iostream>
#include <math.h>
#include <cstdlib>
#include <vector>
#include <complex>

#define epsilon 0.0000001

using namespace std;



complex<double> HornerMethod(vector<complex<double>> a, complex<double> x){

    complex<double> result = a[0];

    for(int i = 1; i < a.size(); i++){
        result = result*x + a[i];
    }

    return result;
}

vector<complex<double>> deflation(complex<double> root, vector<complex<double>> a){

    vector<complex<double>> newMultimian;
    root = -root;
    newMultimian.push_back(a[0]);

    for (int i = 1; i < a.size() - 1; i++) {
```

```
        complex<double> bi = a[i] - (root)*newMultimian[i - 1];
        newMultimian.push_back(bi);
    }

return newMultimian;
}



vector<complex<double>> Derivative(vector<complex<double>> multimian){
    vector<complex<double>> a;
    int k = multimian.size();
    complex<double> n((double)k);
    if(multimian.size() == 1){
        a.push_back(multimian[0]);
        return a;
    }

    for(int i = 0; i < k - 1; i++){
        a.push_back(multimian[i]*(n.real() - 1 - i));
    }

    return a;
}

complex<double> LaguerreMethod(vector<complex<double>> &a, complex<double> x){
    complex<double> z, x0;
    vector<complex<double>> firstDerivative;
    vector<complex<double>> secondDerivative;
    int f = a.size() - 1;
    complex<double> n(a.size() - 1);
    complex<double> upper, lowerPlus, lowerMinus;
```

```
    firstDerivative = Derivative(a);
    secondDerivative = Derivative(firstDerivative);



    for(int i = 0; i < 100000; i++){
        complex<double> value = HornerMethod(a, x);

        if(abs(value) < epsilon) break;

        complex<double> firstDerivativeValue = HornerMethod(firstDerivative, x);
        complex<double> secondDerivativeValue = HornerMethod(secondDerivative, x);

        upper = (n*HornerMethod(a, x));
        lowerPlus = firstDerivativeValue + sqrt((n - 1.0)*((n -
1.0)*firstDerivativeValue*firstDerivativeValue - n*value*secondDerivativeValue));
        lowerMinus = (firstDerivativeValue - sqrt((n - 1.0)*((n -
1.0)*firstDerivativeValue*firstDerivativeValue - n*value*secondDerivativeValue)));

        if(abs(lowerPlus) > abs(lowerMinus)){
            x0 = (upper / lowerPlus);
        }else{
             x0 = (upper / lowerMinus);
         }

        x -= x0;
        if(abs(x0) < epsilon) break;
    }

    return x;
}

vector<complex<double>> FindRoots(vector<complex<double>> a, complex<double> x0){
```

```
    vector<complex<double>> roots;
    vector<complex<double>> activeVector;
    vector<complex<double>> tmpVector;
    activeVector = a;
    complex<double> z, value;
    z = x0;

    for( int i = 0; i < a.size() - 1; i++){
        z = LaguerreMethod(activeVector, x0);
        value = LaguerreMethod(a, z);
        roots.push_back(value);
        tmpVector = deflation(value, activeVector);
        activeVector = tmpVector;
    }

    return roots;
}

void DisplayMZ(vector<complex<double>> roots){
    for (int i = 0; i < roots.size(); i++) {
        if(abs(roots[i].real()) < epsilon*1000)
            roots[i].real(0);
        if(abs(roots[i].imag()) < epsilon*1000)
            roots[i].imag(0);
        cout << roots[i] << " ";
    }cout << endl << endl;
}




int main(){
```

```
//243z^7 − 486z^6 + 783z^5 − 990z^4 + 558z^3 − 28z^2 − 72z + 16 = 0

vector<complex<double>> a1 = {243.0, -486.0, 783.0, -990.0, 558.0, -28.0, -72.0, 16.0};
vector<complex<double>> a2 = {1.0, 1.0, 3.0, 2.0, -1.0, -3.0, -11.0, -8.0, -12.0, -4.0, -4.0};
vector<complex<double>> a3 = {1.0, {0.0, 1.0}, -1.0, {0.0, -1.0}, 1.0};
// vector<complex<double>> a = {1.0, 12.0, 58.0, 134.0, 146.0, 60.0};
complex<double> z, value;

vector<complex<double>> roots;



complex<double> x1 = 1.0;
complex<double> x2 = -3.0;
complex<double> x3 = -1.0;

cout << "Równanie dla którego liczymy:\n";
for(int i = 0; i < a1.size(); i++)
   cout << a1[i] << " ";
cout << endl;
roots = FindRoots(a1, x1);
DisplayMZ(roots);

cout << "Równanie dla którego liczymy:\n";
for(int i = 0; i < a2.size(); i++)
   cout << a2[i] << " ";
cout << endl;
roots = FindRoots(a2, x2);
DisplayMZ(roots);

cout << "Równanie dla którego liczymy:\n";
```

```
    for(int i = 0; i < a3.size(); i++)
        cout << a3[i] << " ";
    cout << endl;
    roots = FindRoots(a3, x3);
    DisplayMZ(roots);



    return 0;
}
```