

Karol Kosoń

Zadanie 11

Zadanie 11

Rozważmy równanie

$$(x^2 - 1) \sinh^3 x = 0. \quad (1)$$

Stosując algorytm siecznych oraz algorytm oparty o trzypunktową interpolację odwrotną proszę znaleźć rozwiązania równania (1), startując, odpowiednio, z dwóch i trzech losowych punktów z przedziału (0,1). W przypadku metody siecznych punkty początkowe mają być dwoma z trzech punktów początkowych dla algorytmu opartego o iterację odwrotną. Miejsce zerowe proszę wyznaczyć z dokładnością 10^{-8} . Zadanie należy powtórzyć dla kilkunastu różnych zestawów punktów początkowych. Proszę przedyskutować otrzymane wyniki.

Metodę siecznych oparłem o ten wzór:

$$x_3 = \frac{f(x_1)x_2 - f(x_2)x_1}{f(x_1) - f(x_2)}.$$

Wyliczam kolejne miejsca zerowe i przesuwam punkty. Iteracja działa dopóty, dopóki różnica otrzymanych miejsc zerowych jest większa od epsilon.

Metodę interpolacji odwrotnej oparłem o ten wzór:

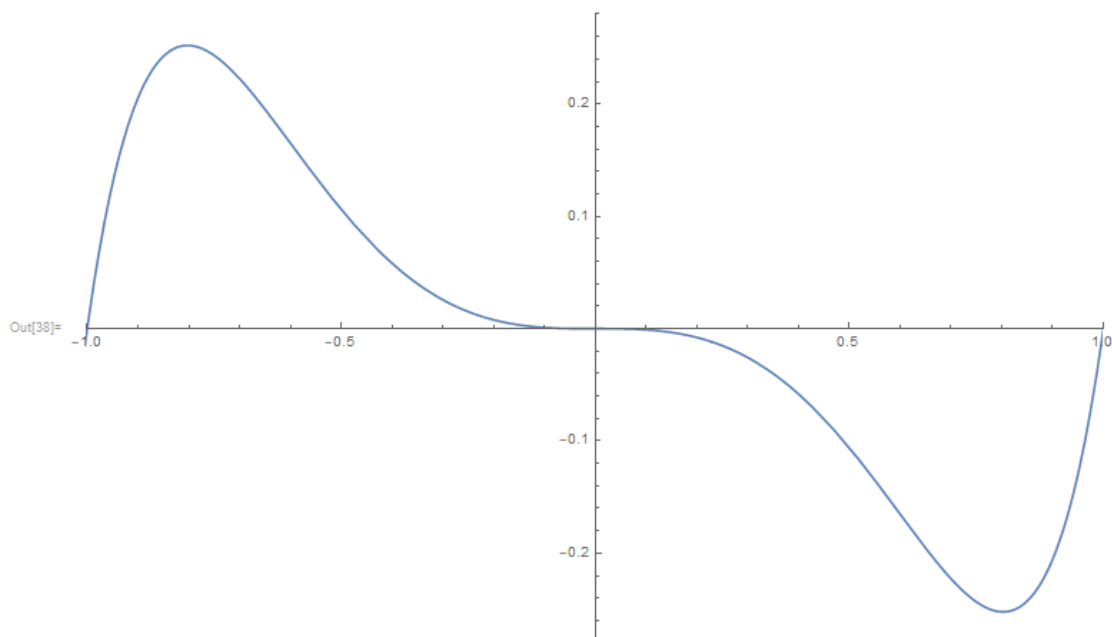
$$x_{n+1} = \frac{f_{n-1}f_n}{(f_{n-2} - f_{n-1})(f_{n-2} - f_n)}x_{n-2} + \frac{f_{n-2}f_n}{(f_{n-1} - f_{n-2})(f_{n-1} - f_n)}x_{n-1} + \frac{f_{n-2}f_{n-1}}{(f_n - f_{n-2})(f_n - f_{n-1})}x_n,$$

Wyliczam, kolejne x-y i wyznaczam wartości funkcji w tym punkcie. Po każdej iteracji punkty się przesuwają zgodnie z wyliczonym x-em.

In[17]:= $f = (x^2 - 1) * \sinh[x]^3$
[sinus hiperbol]

Out[17]:= $(-1 + x^2) \sinh(x)^3$

In[38]:= $\text{Plot}[f, \{x, -1.1, 1.1\}, \text{PlotRange} \rightarrow \{\{-1, 1\}, \{-0.28, 0.28\}\}]$
[wykres] [zakres wykresu]



Widzimy na powyższym wykresie, że funkcja (1), ma miejsca zerowe w $-1, 0$ i 1 . Dla naszego przedziału $(0,1)$, otrzymamy jedynie rozwiązania dla 0 i 1 . Trzeba byłoby przesunąć punkty początkowe bliżej -1 .

Kilka losowych wyników (generuję 3 losowe liczby zmiennoprzecinkowe i wrzucam je do funkcji):

Metoda siecznych	Metoda interpolacji odwrotnej
Ilość iteracji: 2 Miejsce zerowe jest w: 0.6567620852295133	Ilość iteracji: 10 Miejsce zerowe jest w: -0.0000000000000001

$x_1 = 0.6567620852295133$ $x_2 = 0.9041111953110020$ $x_3 = 0.5535490985743464$

Metoda siecznych	Metoda interpolacji odwrotnej
Ilość iteracji: 54 Miejsce zerowe jest w: 0.0000000241318722	Ilość iteracji: 8 Miejsce zerowe jest w: 0.0000000002787713

$x_1 = 0.9667610530586732$ $x_2 = 0.0886938297602785$ $x_3 = 0.4674769562983312$

Metoda siecznych	Metoda interpolacji odwrotnej
Ilość iteracji: 56 Miejsce zerowe jest w: 0.0000000267810923	Ilość iteracji: 8 Miejsce zerowe jest w: 0.0000000019546426

$x_1 = 0.2767600208878331$ $x_2 = 0.2732764646752162$ $x_3 = 0.3814048135566547$

Metoda siecznych	Metoda interpolacji odwrotnej
Ilość iteracji: 10 Miejsce zerowe jest w: 0.9999999999999983	Ilość iteracji: 9 Miejsce zerowe jest w: -0.000000000338307

x1 = 0.8967579570118142 x2 = 0.6424417345050917 x3 = 0.2092605280733018

Metoda siecznych	Metoda interpolacji odwrotnej
Ilość iteracji: 56 Miejsce zerowe jest w: 0.0000000251032760	Ilość iteracji: 9 Miejsce zerowe jest w: 0.0000000229935867

x1 = 0.2067569248409741 x2 = 0.8270243694200294 x3 = 0.1231883853316253

Metoda siecznych	Metoda interpolacji odwrotnej
Ilość iteracji: 56 Miejsce zerowe jest w: 0.0000000290784181	Ilość iteracji: 10 Miejsce zerowe jest w: -0.0000000000004729

x1 = 0.8267548614306165 x2 = 0.1961896387842436 x3 = 0.9510441003139336

Metoda siecznych	Metoda interpolacji odwrotnej
Ilość iteracji: 58 Miejsce zerowe jest w: 0.0000000246446886	Ilość iteracji: 9 Miejsce zerowe jest w: -0.0000000335831454

x1 = 0.4467527970889363 x2 = 0.5653549086141190 x3 = 0.7788998148305807

Metoda siecznych	Metoda interpolacji odwrotnej
Ilość iteracji: 53 Miejsce zerowe jest w: - 0.0000000255459981	Ilość iteracji: 7 Miejsce zerowe jest w: -0.0000015297217824

x1 = 0.7567517653837575 x2 = 0.7499375435290567 x3 = 0.6928276720889042

Metoda siecznych	Metoda interpolacji odwrotnej
Ilość iteracji: 57 Miejsce zerowe jest w: 0.0000000261937434	Ilość iteracji: 8 Miejsce zerowe jest w: 0.0000000522255630

x1 = 0.3067466049952184 x2 = 0.6728507176380840 x3 = 0.2624669583805217

Dołożyłem również punkty blisko -1, żeby udowodnić, że metoda działa:

Metoda siecznych	Metoda interpolacji odwrotnej
Ilość iteracji: 7 Miejsce zerowe jest w: - 1.0000000000000000	Ilość iteracji: 12 Miejsce zerowe jest w: -0.0000000000011477

x1 = -0.9600000000000000 x2 = -0.7600000000000000 x3 = -0.6500000000000000

KOD:

```
#include <iostream>
```

```
#include <iomanip>
```

```
#include <cmath>
```

```
#include <cstdlib>
```

```
#include <ctime>
```

```
using namespace std;
```

```
const double epsilon = 0.00000001;
```

```
double f(double x)
```

```
{
```

```
    return (x*x - 1)*(sinh(x)*sinh(x)*sinh(x));
```

```
}
```

```
double secantMethod(double x1, double x2){
```

```
    double f1, f2, f3, x3, tmpX3;
```

```
    int i = 0;
```

```
    while(1)
```

```
{
```

```
    f1 = f(x1);
```

```
    f2 = f(x2);
```

```
    x3 = (f1 * x2 - f2 * x1)/( f1 - f2);
```

```
    f3 = f(x3);
```

```
    if(f1 == f2 )
```

```
{
```

```
    cout << "Złe punkty startowe, f1 = f2\n";
```

```
    break;
```

```
}
```

```

if(abs(x3 - tmpX3) <= epsilon) break;

cout << "W x3 = "<< x3 << ", f(x3) = " << f3 << endl;

x2 = x1;

x1 = x3;

tmpX3 = x3;


i++;

}


cout << "Ilość iteracji: " << i << endl;


return x3;
}


double InverseInterpolation(double x1, double x2, double x3){

    double x0, y;

    double tmpX = 0;

    double x[] = {x1, x2, x3};

    double fx[] = {f(x1), f(x2), f(x3)};

    int i = 0;


    while(1){

        i++;

        x0 = (x[0]*fx[1]*fx[2])/((fx[0] - fx[1] * (fx[0] - fx[2]))) + (x[1]*fx[0]*fx[2])/((fx[1] - fx[0] * (fx[1] -
fx[2])))
        + (x[2]*fx[0]*fx[1])/((fx[2] - fx[0] * (fx[2] - fx[1])));

        x1 = x2;

        x2 = x3;

        x3 = x0;

        y = f(x0);

```

```

x[0] = x1;
fx[0] = f(x1);
x[1] = x2;
fx[1] = f(x2);
x[2] = x0;
fx[2] = f(x0);
cout << "W x = " << x0 << ", f(x) = " << f(x0) << endl;

if(abs(x0 - tmpX) <= epsilon){
    cout << "W x = " << x0 << ", f(x) = " << f(x0) << endl;

    i++;
    break;
}

// if(abs(f(x0)) <= epsilon){
//     cout << "W x = " << x0 << ", f(x) = " << f(x0) << endl;
//     i++;
//
//     break;
// }

tmpX = x0;
}

cout << "Ilość iteracji: " << i << endl;

return x0;
}

double fRand(double fMin, double fMax)
{
    double f = (double)rand() / RAND_MAX;

```

```
    return fMin + f * (fMax - fMin);  
}
```

```
int main()
```

```
{
```

```
    double x0,f0,f1,f2;
```

```
    double value;
```

```
    srand(time(0));
```

```
    double x1 = fRand(0,1);
```

```
    double x2 = fRand(0,1);
```

```
    double x3 = fRand(0,1);
```

```
    // double x1 = -.96;
```

```
    // double x2 = -.76;
```

```
    // double x3 = -.65;
```

```
    cout << setprecision(16)    // 8 cyfr po przecinku
```

```
        << fixed;           // format stałoprzecinkowy
```

```
    cout << "x1 = " << x1 << " x2 = " << x2 << " x3 = " << x3 << endl;
```

```
    value = secantMethod(x1, x2);
```

```
    cout << "Miejsce zerowe jest w: " << value << endl;
```

```
    value = InverseInterpolation(x1, x2, x3);
```

```
    cout << "Miejsce zerowe jest w: " << value << endl;
```

```
    return 0;
```

```
}
```