METODY NUMERYCZNE

ZADANIE 7

Artur Guniewicz

7 O. Znajdź, z dokładnością do czterech cyfr dziesiętnych, wartości współczynników wielomianu interpolacyjnego opartego na następującej tabelce:

x	0.062500	0.187500	0.312500	0.437500	0.562500	0.687500	0.812500	0.937500
$f(x)$	0.687959	0.073443	−0.517558	−1.077264	−1.600455	−2.080815	−2.507266	−2.860307

Sporządź wykres uzyskanego wielomianu w przedziale $-1 \leqslant x \leqslant 1$ i zaznacz na nim punkty, które posłużyły do jego konstrukcji.

Metoda: macierz Vandermonde'a

Metoda ta pozwala bezpośrednio wyznaczyć współczynniki wielomianu interpolacyjnego.

$$\begin{bmatrix} x_1^{n-1} & x_1^{n-2} & \cdots & x_1 & 1 \\ x_2^{n-1} & x_2^{n-2} & \cdots & x_2 & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_n^{n-1} & x_n^{n-2} & \cdots & x_n & 1 \end{bmatrix} \begin{bmatrix} a_{n-1} \\ a_{n-2} \\ \vdots \\ a_0 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix}$$

Macierz ta powinna przejść faktoryzację LU, a następnie można obliczyć współczynniki wektora a.

Kod programu:

```
/*
************************************************************
*                                                          *
*               Artur Guniewicz - Zadanie 7                *
*                                                          *
************************************************************
*/

#include <iostream>
#include <iomanip>
#include <fstream>
#include <stdio.h>
#include <stdlib.h>
#include <gsl/gsl_math.h>
#include <gsl/gsl_matrix.h>
#include <gsl/gsl_linalg.h>
#include <gsl/gsl_eigen.h>
#include <gsl/gsl_sf.h>

using namespace std;

const int N = 8;

int main()
{
   int s;

   double data_x[] = {0.062500, 0.187500, 0.312500, 0.437500, 0.562500,
0.687500, 0.812500, 0.935700};
      double  data_fx[]  =  {0.687959,  0.073443,  -0.517558,  -1.077264,
-1.600455, -2.080815, -2.507266, -2.860307};

   gsl_vector_view x = gsl_vector_view_array(data_x, N);
   gsl_vector_view b = gsl_vector_view_array(data_fx, N);

   gsl_vector *a = gsl_vector_alloc(N);
   gsl_permutation *p = gsl_permutation_alloc(N);
   gsl_matrix *V = gsl_matrix_alloc(N, N);
```

```
   // w macierzy V na pozycji (i,j) ustawiona zostaje wartość wzięta z
wektora x z pozycji i
   // podniesiona do potęgi N-1-j
   for (int i = 0; i < N; i++)
       for (int j = 0; j < N; j++)
                                              gsl_matrix_set(V,    i,    j,
gsl_sf_pow_int(gsl_vector_get(&x.vector, i), N - 1 - j));

   gsl_linalg_LU_decomp(V, p, &s); // faktoryzacja LU
     gsl_linalg_LU_solve(V, p, &b.vector, a); // rozwiązanie równania
Ax=b

    cout << setprecision(4) << fixed; // precyzja z dokładnością do 4
cyfr po przecinku
   for (int i = 0; i < N; i++)
        cout << "a" << N - 1 - i << " = " << gsl_vector_get(a, i) <<
endl;

   cout << endl;
   cout << "Wielomian interpolacyjny ma postać:" << endl;
   cout << "w(x) = ";
   for (int i = 0; i < N; i++)
   {
       if (i != N - 1)
            cout << gsl_vector_get(a, i) << "x^" << N - 1 - i << "   +
";
       else
           cout << gsl_vector_get(a, i) << "x^" << N - 1 - i << endl;
   }

   gsl_matrix_free(V);
   gsl_vector_free(a);

   return 0;
}
```

Kompilacja:

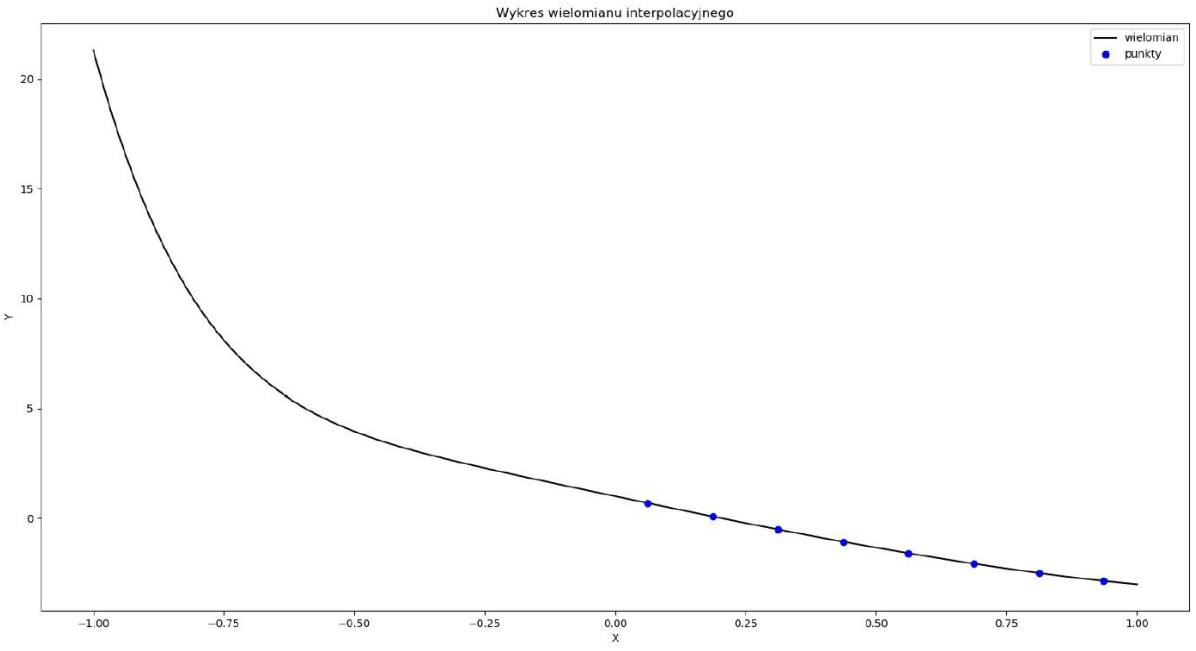
g++ Zadanie7.cpp -ansi -pedantic -Wall -I/<ścieżka dostępu do biblioteki gls> -lgsl -lgslcblas -lm -o Zadanie7 && ./Zadanie7

Wyniki:

```
a7 = -1.9207
a6 = 5.8857
a5 = -6.3099
a4 = 2.7206
a3 = 0.3130
a2 = 0.3262
a1 = -5.0304
a0 = 1.0010

Wielomian interpolacyjny ma postać:
w(x) = -1.9207x^7  +  5.8857x^6  +  -6.3099x^5  +  2.7206x^4  +  0.3130x^3  +  0.3262x^2  +  -5.0304x^1  +  1.0010x^0
```

Wykres:
Stworzony za pomocą gnuplot'a.



Komentarz:

W programie zastosowałem funkcje z biblioteki GSL. Jest to bardzo szybki i efektowny (przez co bardzo dobry) sposób.

Dokumentacja ww biblioteki:
https://www.gnu.org/software/gsl/doc/html/index.html
Działania ważniejszych funkcji opisane w komentarzach w kodzie programu.